

Raciocínio Baseado Em Casos

Case-based reasoning

CBR

Engenharia Informática
Inteligência Artificial
Eduardo J. Solteiro Pires

Autor

Marcelo Queirós Pinto – A160102

Vila Real, 2016

Resumo

Um Sistema Baseado em Casos é um tipo de sistema que utiliza Raciocínio Baseado em Casos para resolver problemas, o seu funcionamento recorre ao processo de resolver novos problemas baseando-se em soluções de problemas similares anteriores, podendo ainda haver uma adaptação das soluções ao caso específico, mas partindo sempre de um ou mais caso(s) anterior(es).

O tema que trata este trabalho é a avaliação de telemóveis usados, o qual se revela importante para qualquer pessoa que deseje vender o seu telemóvel, seja na internet, pessoalmente ou numa loja. Um telemóvel, na atualidade, sofre de um decaimento muito rápido do seu valor, por conta das constantes inovação tecnológicas, das exigências do mercado, que procura sempre novos produtos, os quais são lançados muito rápido, e por conta do uso constante por parte do utilizador. Estes parâmetros influenciam o valor de um telemóvel, o que levam a que o utilizador tenha pouca noção do valor real do seu telemóvel.

Índice

| | |
|--|----|
| 1. Introdução | 5 |
| 2. Estado da Arte | 6 |
| 3. Descrição do modelo e funções adotadas | 11 |
| 3.1. Função de Recuperação e Reutilização | 12 |
| 3.2. Função de Retensão | 14 |
| 3.3. Função de Revisão | 15 |
| 3.4. Funções de similaridade | 23 |
| 4. Implementação | 33 |
| 4.1. Diagrama SQL | 33 |
| 4.2. Views | 34 |
| 5. Conclusão | 39 |
| 6. Bibliografia/Referências | 40 |
| 7. Anexos | 42 |
| 7.1. Asp.Net | 42 |
| 7.2. Código SQL – Definição da Base de Dados | 86 |
| 7.3. Código SQL – Dados | 89 |

Índice de Figuras

| | |
|--|----|
| Figura 1: Funcionamento de um Sistema Baseado em Casos | 7 |
| Figura 2: Diagrama da Base de Dados | 33 |
| Figura 3: View Avaliação..... | 34 |
| Figura 4: View Avaliação - Parte 2 | 34 |
| Figura 5: View Valor da Avaliação | 35 |
| Figura 6: View da Base de Casos | 36 |
| Figura 7: View Adicionar Caso | 36 |
| Figura 8: View Editar Caso | 37 |
| Figura 9: View Confirmação de eliminação de Caso | 37 |
| Figura 10: View Sobre..... | 38 |
| Figura 11: View Contactos | 38 |

1. Introdução

No âmbito da Unidade Curricular de Inteligência Artificial, foi proposto o presente trabalho, cujo tema é a implementação de Raciocínio Baseado em Casos em um problema à escolha de cada aluno (ou grupo de alunos).

O tema que trata este trabalho é a avaliação de telemóveis usados, o qual se revela importante para qualquer pessoa que deseje vender o seu telemóvel, seja na internet, pessoalmente ou numa loja. Um telemóvel, na atualidade, sofre de um decaimento muito rápido do seu valor, por conta das constantes inovação tecnológicas, das exigências do mercado, que procura sempre novos produtos, os quais são lançados muito rápido, e por conta do uso constante por parte do utilizador. Estes parâmetros influenciam o valor de um telemóvel, o que levam a que o utilizador tenha pouca noção do valor real do seu telemóvel.

O que se propõe neste trabalho é o desenvolvimento de um *site*, que possibilite a qualquer pessoa, a rápida consulta do valor de mercado do seu telemóvel, inserindo algumas características de rápida resposta (no total 12 características analisadas).

Os algoritmos utilizados decorrem dos conhecimentos obtidos nas aulas teóricas e práticas de Inteligência Artificial acerca do Raciocínio Baseado em Casos, assim como estudo e pesquisa mais aprofundados sobre o tema por parte do autor deste trabalho.

O Raciocínio Baseado em Casos é um método que se revela muito útil em várias situações, como por exemplo, quando os requisitos de um problema não são totalmente conhecidos. Assim, se conhecermos casos acerca do problema em questão, uma maneira inteligente de o abordar é a comparação com os casos que conhecemos, chegando a uma conclusão para o novo caso.

O desenvolvimento deste relatório será feito a partir de princípios hermenêuticos, baseando-se na revisão bibliográfica, código efetuado e conhecimentos adquiridos.

Para a realização deste trabalho foi utilizado o *software Visual Studio Enterprise 2015* para o desenvolvimento de código e conexão à base de dados, a qual foi elaborada no *software MS SQL Server*. A linguagem utilizada para elaboração do código funcional foi *Asp.Net*.

2. Estado da Arte

Um Sistema Baseado em Casos é um tipo de sistema que utiliza Raciocínio Baseado em Casos para resolver problemas, ou seja, o seu funcionamento recorre ao processo de resolver novos problemas baseando-se em soluções de problemas similares anteriores, podendo ainda haver uma adaptação das soluções ao caso específico, mas partindo sempre de um caso anterior.

O sistema parte sempre de um conjunto de casos que conhece, e ainda extrai conhecimento a partir de novos casos, adicionando-os à base de casos, depois de encontrar uma solução adequada para o problema.

Segundo os investigadores na área da inteligência artificial, entre outras, o raciocínio baseado em casos define-se como um “método de desenvolvimento de um sistema especialista baseado em conhecimento que desempenha exemplos de experiências passadas num domínio específico que pode ser aplicado em novos problemas” (DUTA, 1993), ou, mais extensamente, como “um raciocínio baseado na memória de experiências prévias. Um raciocinador utilizando experiências antigas (casos) poderá utilizar esses casos para sugerir soluções para problemas, para apontar potenciais novos problemas com uma solução a ser computada, para interpretar uma nova situação e fazer previsões acerca do que poderá acontecer, ou criar argumentos que justifiquem algumas conclusões” (Kolodner, 1993).

Num estudo sobre diagnóstico nutricional, a autora Maria Alice Lagos Thé (2001), refere que “uma vantagem fornecida por sistemas inteligentes que utilizam raciocínio baseado em casos é a habilidade de estocar conhecimentos de vários especialistas, tornando a tarefa do investigador representar mais a realidade e ser propenso a menores erros”. Nesta investigação, a partir da validação do modelo *Fuzzy* e da metodologia de RBC, a autora conseguiu a implementação da tarefa nutricional próxima dos 100% com a entrada de um novo caso na base de casos. Mencionando Watson (1997), a autora refere que “quando empregado o paradigma de relembrar o prévio episódio mais similar, os humanos são sujeitos a colocar emoções que podem evitar recuperação relevante das soluções, a passo que um sistema computacional, logo que realiza requerimentos teóricos, garante que as similaridades das experiências sejam recuperadas”.

Este tipo de sistema tem grande aplicabilidade, porque permite a obtenção de boas soluções tendo-se pouco domínio do problema, podendo propor soluções em casos em que é desconhecido o estado completo do problema. Chega-se à solução apenas a partir do conhecimento adquirido de

outros casos, espera-se assim uma melhoria no desempenho quando o número de casos da base de casos aumenta.

Apresenta-se uma figura ilustrativa do funcionamento de um sistema deste tipo:

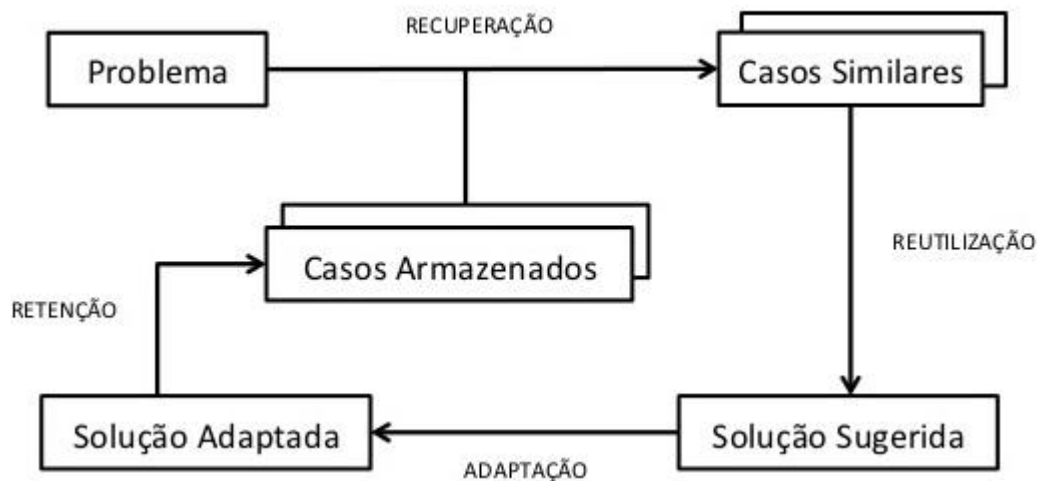


Figura 1: Funcionamento de um Sistema Baseado em Casos

De seguida, serão explicadas as etapas ilustradas na figura, descrevendo estas o funcionamento ao qual recorre um Sistema Baseado em Casos.

Recuperação

Dada uma descrição de um problema, ou simplesmente as características de um problema, esta etapa é responsável por encontrar o caso ou casos mais similares à situação atual, utilizando os índices da Base de Casos, um identificador único. Esta etapa baseia-se nos índices e na organização de memória para dirigir a procura dos casos potencialmente úteis, ou do caso mais similar.

Os algoritmos mais simples baseiam-se no exame exaustivo das características dos casos e recuperam o caso com o maior número de características idênticas. Outros buscam o melhor caso (Kolodner, 1989), utilizando-se de heurísticas para reduzir e dirigir a procura. Entre esses pode-se citar: busca serial (Simoudis et al, 1993), busca hierárquica (Maher e Zhang 1991) e busca por simulação paralela (Domeshek 1993).

Entre os métodos mais conhecidos de recuperação de casos estão o algoritmos de vizinhança, de indução, indução guiada por conhecimento e recuperação de padrões, que são utilizados sozinhos ou combinados.

Reutilização

A partir do caso recuperado é feita a reutilização da solução associada àquele caso. A solução do caso recuperado pode ser transferida para o novo problema diretamente como sendo a sua solução, ou pode ser efetuada uma adaptação/revisão da solução.

Adaptação/Revisão

Quando uma situação é fornecida, o algoritmo de recuperação traz o caso mais similar que encontra. Normalmente, o caso selecionado não é perfeitamente igual à descrição do problema do utilizador. Caso existam diferenças entre o problema do utilizador e o caso devolvido na etapa recuperação estas devem ser tidas em conta. O processo de adaptação procura diferenças salientes entre as duas descrições e aplica regras de forma a ajustar a solução

Embora as regras aparentem ser bastante específicas, construí-las para adaptação é muito mais simples do que desenvolver um sistema puramente baseado em regras, desde que os casos armazenados tenham uma razoável cobertura sobre o domínio do problema. A adaptação é feita a partir de um conjunto menor de regras, resultando em uma maior eficiência e eficácia na obtenção da solução ideal.

Retensão/Aprendizagem

Processo final, posterior ao arranjo da solução para o problema em questão. O objetivo é armazenar o novo caso e a sua respectiva solução para futuras recuperações. O sistema irá decidir que informações armazenar e de que forma.

Além dos trabalhos e autores referidos acima, serão referidos abaixo outros trabalhos, será dada uma explicação acerca do que foi feito em cada trabalho mencionado. Nem todos os trabalhos estão diretamente relacionados com o tema, uma vez que após pesquisa exaustiva, foram encontrados vários trabalhos mas poucos relacionados com o tema do presente trabalho.

Forecasting of manufacturing cost in mobile phone products by case-based reasoning and artificial neural network models

Autores:

1. *Department of Information Management Yuan Ze University Taoyuan Taiwan, R.O.C*

2. *Department of Naval Architecture National Kaohsiung Marine University Kaohsiung Taiwan*

Descrição:

Os fabricantes de telemóveis em Taiwan fizeram grandes esforços em propor as cotações racionais para as empresas de telemóveis internacionais com a ambição de ganhar a outros fabricantes. No entanto, há várias incertezas e problemas a serem resolvidos na estimativa dos custos de fabricação para os fabricantes de telefones. Não existe um modelo existente que possa ser aplicado diretamente na previsão dos custos de produção. Este trabalho faz a primeira tentativa de desenvolver um sistema híbrido de Raciocínio Baseado em Casos e Redes Neurais Artificiais como um modelo de previsão de custo unitário de produto para a *Mobile Phone Company*. De acordo com a fórmula de custo do telemóvel e as opiniões dos peritos, um conjunto de fatores qualitativos e quantitativos são analisados e determinados. Os fatores qualitativos são aplicados na Sistema Baseado em Casos para recuperar um caso semelhante a partir da base de casos para um novo produto de telemóvel e as redes neurais são usadas para encontrar a relação entre os fatores quantitativos e a previsão de custo unitário de produto prevista. Finalmente, experiências intensivas são conduzidas para testar a eficácia de seis diferentes modelos de previsão. O modelo proposto neste trabalho é comparado com os outros cinco modelos e o valor de *MAPE* do modelo proposto é o menor. Esta pesquisa fornece um novo modelo de previsão com alta precisão para empresas de fabricação de telemóveis.

CookIIS Mobile: A Case-Based Reasoning Recipe Customizer for Android Phones

Autores:

Kerstin Bach, Klaus-Dieter Althoff, Julian Satzky, and Julian Kroehl (Competence Center Case-Based Reasoning, German Research Center for Artificial Intelligence (DFKI) GmbH) - University of Hildesheim Institute of Computer Science - Intelligent Information Systems Lab

Descrição: *CookIIS* é uma aplicação Android que permite a gestão de receitas, permitindo ao utilizador inserir as receitas que faz diariamente. Após um período inicial de recolha de casos, o software consegue sugerir receitas ao utilizador, tendo em conta o seu gosto pessoal e as receitas que o mesmo costuma fazer.

Case-Based Reasoning for Selecting Study Program in Senior High School

Autores:

Sri Mulyana, Sri Hartati, Retantyo Wardoyo, Edi Winarko (Department of Computer Sciences and Electronics) - Gadjah Mada University

Descrição:

Esta aplicação é utilizada para auxiliar os alunos na seleção do programa de estudo. Os casos utilizados no estudo incluem resultados do teste de inteligência, interesse do aluno e notas de vários assuntos. Os casos com maior grau de similaridade são recomendados como soluções.

Case-based reasoning combined with statistics for diagnostics and prognosis

Autores:

T. Olsson and P. Funk

Descrição:

Hoje em dia, muitas abordagens utilizadas para diagnóstico são baseadas em um modelo preciso. Isso exclui o diagnóstico de muitos tipos complexos de máquinas que não podem ser modeladas e simuladas facilmente ou sem grande esforço. O objetivo deste *software* foi mostrar que, ao incluir a experiência humana, é possível diagnosticar máquinas complexas quando não há modelos limitados ou simulações disponíveis. Isso também permite o diagnóstico em uma aplicação dinâmica onde as condições mudam e novos casos são frequentemente adicionados. Assim, cada novo caso resolvido aumenta o poder de diagnóstico do sistema.

3. Descrição do modelo e funções adotadas

O tema que este trabalho trata é a avaliação de telemóveis usados, o qual se revela um tema em que a solução pode ser bastante complicada de encontrar, uma vez que um telemóvel sofre um decaimento do seu valor logo após a sua compra, por conta das constantes inovação tecnológicas, o lançamento rápido de novos produtos, e por conta do uso constante por parte do utilizador, que dará uso ao telemóvel. Todos estes parâmetros influenciam o valor de um telemóvel, além do suas características inerentes, e portanto, houve um especial cuidado na escolha das variáveis, nos pesos que lhe são atribuídos, na escolha dos valores de similaridade entre os atributos e nas funções de adaptação.

As características pedidas ao utilizador são as seguintes:

- Marca (Peso=0,35);
- Memória RAM (dada em *GBs*, Peso=0,075);
- Memória interna (dados em *GBs*, Peso=0,05);
- Potência da Bateria (dada em *mAh*, Peso=0,05);
- Resolução do Ecrã (Peso=0,1);
- Tamanho do Ecrã(Peso=0,025);
- Velocidade do processador (dada em *GHz*, Peso=0,1);
- Número de núcleos do processador(Peso=0,025);
- Resolução da câmara frontal (dada em *megapixels*, Peso=0,025);
- Resolução da câmara traseira (dada em *megapixels*, Peso=0,05);
- Estado do telemóvel(Peso=0,1);
- Idade do telemóvel (dada em meses, Peso = 0,05);

3.1. Função de Recuperação e Reutilização

A partir do conjunto das características acima, utilizando uma função de similaridade individual para cada uma com retorno normalizado de 0 a 1, é feita a etapa recuperação numa função que chama todas as funções de similaridade e calcula a similaridade do telemóvel inserido com cada telemóvel da base de casos, atribuindo ao retorno de cada função de similaridade, os pesos acima indicados. O telemóvel com maior índice de similaridade é o escolhido. O valor do telemóvel inserido ficara com o valor do caso com mais similaridade. Depois, haverá a fase adaptação para ajustar o valor do telemóvel, se este não for exatamente igual ao caso com mais similaridade. De seguida, apresenta-se a função descrita:

```
public void Avaliação()
{
    double melhorSimilaridade = 0; //ainda não foi encontrado nenhuma aproximação
    double similaridadeItem = 0;
    int idItem = 0;

    const double pesoMarca = 0.35;
    const double pesoRam = 0.075;
    const double pesoMemoriaInterna = 0.05;
    const double pesoAhBateria = 0.05;
    const double pesoResEcra = 0.1;
    const double pesoTamanho = 0.025;
    const double pesoProcessadorVelocidade = 0.1;
    const double pesoProcessadorNucleos = 0.025;
    const double pesoCameraFrontal = 0.025;
    const double pesoCameraTraseira = 0.05;
    const double pesoEstado = 0.1;
    const double pesoIdade = 0.05;

    foreach (Telemovel item in DataBase.Telemovels) //RECUPERAÇÃO
    {
        //pesos acima
        similaridadeItem = SimilaridadeMarca(DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome,
item.marca.nome) * pesoMarca + SimilaridadeRam(tm.ram, item.ram) * pesoRam +
SimilaridadeMemoriaInterna(tm.memoriaInterna, item.memoriaInterna) * pesoMemoriaInterna +
```

```

SimilirdademAhBateria(tm.mAhBateria, item.mAhBateria) * pesomAhBateria +
SimilirdadeResoluçãoeCra(DataBase.ResoluçãoeCras.Single(x => x.idResoluçãoeCra ==
e.idResoluçãoeCra).designação, item.Ecra.ResoluçãoeCra.designação) * pesoResEcra +
SimilirdadeTamanhoEcra(e.tamanho, item.Ecra.tamanho) * pesoTamanho +
SimilirdadeVelocidadeProcessador(p.velocidadeProcessador, item.Processador.velocidadeProcessador) *
pesoProcessadorvelocidade + SimilirdadeNucleosProcessador(p.nucleosProcessador,
item.Processador.nucleosProcessador) * pesoProcessadornucleos +
SimilirdadeResolucaoCameraFrontal(c.resolucaoFrontal, item.Camera.resolucaoFrontal) * pesoCamerafrontal +
SimilirdadeResolucaoCameraTraseira(c.resolucaoTraseira, item.Camera.resolucaoTraseira) * pesoCameratraseira +
SimilirdadeEstado(DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação, item.Estado.designação)
* pesoEstado + SimilirdadeIdade(tm.idade, item.idade) * pesoIdade;

```

```

    if (similirdadeItem > melhorSimilirdade)
    {
        //REUTILIZAÇÃO
        melhorSimilirdade = similirdadeItem;
        tm.valorFinal = Convert.ToInt32(item.valorFinal); //int da base de dados nao é compativel com int
do visual studio
        idItem = item.idTelemovel;
    }
}
submeterNaDB(); // RETENSÃO

if (similirdadeItem != 1)
    acertarPreço(DataBase.Telemoveis.Single(x => x.idTelemovel == idItem)); //REVISÃO

tm = new Telemovel(); //reinicializar para a proxima avaliação
e = new Ecra();
c = new Camera();
p = new Processador();
}

```

3.2. Função de Retensão

A função de revisão submete as tabelas correspondentes à camara, ecrã e processador na base de dados. Depois há uma associação do telemóvel aos ids correspondentes das tabelas da base de dados. E depois, finalmente, o telemóvel é submetido na base de dados.

É de notar que as tabelas Resolução de ecrã, Marca e Estado não são submetidas porque elas já existem, e as chaves estrangeiras do telemóvel correspondentes a essas tabelas são associadas no decorrer do programa. De seguida, apresenta-se a função de retenção:

```
public void submeterNaDB()
{
    DataBase.Cameras.InsertOnSubmit(c);
    DataBase.Ecras.InsertOnSubmit(e);
    DataBase.Processadors.InsertOnSubmit(p);
    DataBase.SubmitChanges();

    tm.idCameras = c.idCameras;
    tm.idEcra = e.idEcra;
    tm.idProcessador = p.idProcessador;
    DataBase.Telemovels.InsertOnSubmit(tm);
    DataBase.SubmitChanges();
}
```

3.3. Função de Revisão

A função de revisão é executada quando a similaridade entre o caso mais próximo e o telemóvel inserido não é 1, ou seja, não são completamente iguais.

O acerto de valor baseia-se sempre nos mesmos princípios, caso as características sejam *strings*, é verificado cada caso e acertado o valor individualmente, adicionando ou subtraindo um valor multiplicado pelo próprio valor. Este valor foi decidido caso a caso cuidadosamente.

Caso as características sejam valores inteiros ou de virgula flutuante, é utilizada a função de similaridade para alterar pouco o valor quando a similaridade é alta e bastante o valor quando a similaridade é baixa. Para isto é utilizado o contrário do retornado pela função, subtraindo ao valor 1, o que foi retornado pela função. Depois o valor originado ainda é multiplicado por um valor decidido cuidadosamente e após vários testes.

De seguida apresenta-se a função de revisão:

```
public void acertarPreço(Telemovel caso)
{
    List<Telemovel> listT = DataBase.Telemovels.OrderByDescending(p => p.idTelemovel).ToList();
    int idT = listT.First().idTelemovel; //para buscarmos o ultimo a ser inserido (primeiro nesta lista porque
está ordenada em descending)

    if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome != caso.marca.nome)
    {
        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" && caso.marca.nome
== "Samsung") DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal * 0.1);
        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "Apple") DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal * 0.1);
        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" && caso.marca.nome ==
"Samsung") DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal * 0.2);
        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "LG") DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal * 0.2);
    }
```

```

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" && caso.marca.nome ==
"Apple") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.3);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" && caso.marca.nome
== "LG") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.3);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" && caso.marca.nome
== "LG") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.3);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" && caso.marca.nome ==
"Outras") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.3);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" && caso.marca.nome
== "Samsung") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.4);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "Outras") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.4);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" && caso.marca.nome
== "Apple") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.6);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" && caso.marca.nome
== "Outras") DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.6);
    }

    if (tm.ram > caso.ram)
    {
        DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * (1 -
SimiliridadeRam(tm.ram, caso.ram)) * 0.05);
    }

    if (tm.ram < caso.ram)
    {
        DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * (1 -
SimiliridadeRam(tm.ram, caso.ram)) * 0.05);
    }

```



```

    }

    if (tm.memoriaInterna > caso.memoriaInterna)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeMemoriaInterna(tm.memoriaInterna, caso.memoriaInterna)) * 0.025);
    }

    if (tm.memoriaInterna < caso.memoriaInterna)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeMemoriaInterna(tm.memoriaInterna, caso.memoriaInterna)) * 0.025);
    }

    if (tm.mAhBateria > caso.mAhBateria)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidademAhBateria(tm.mAhBateria, caso.mAhBateria)) * 0.05);
    }

    if (tm.mAhBateria < caso.mAhBateria)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidademAhBateria(tm.mAhBateria, caso.mAhBateria)) * 0.05);
    }

    if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao !=
    caso.Ecras.ResolucaoEcras.designacao)
    {
        if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
        "640x480 (VGA)" && caso.Ecras.ResolucaoEcras.designacao == "1280x720 (HD)")
        {
            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
            (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.05);
        }
    }

```

```

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"1280x720 (HD)" && caso.Ecra.ResoluçãoEcra.designação == "640x480 (VGA)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.05);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação == "1920x1080 (FULL HD)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.08);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"1920x1080 (FULL HD)" && caso.Ecra.ResoluçãoEcra.designação == "640x480 (VGA)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.08);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação == "2560x1440 (2K)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.1);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação == "640x480 (VGA)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.1);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação == "3840x2160 (4K)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.15);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação == "640x480 (VGA)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.15);

        //-----

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"1920x1080 (FULL HD)" && caso.Ecra.ResoluçãoEcra.designação == "1280x720 (HD)")
            DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveil == idT).valorFinal * 0.06);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"1280x720 (HD)" && caso.Ecra.ResoluçãoEcra.designação == "1920x1080 (FULL HD)")

```

```

DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.06);

if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"2560x1440 (2K)" && caso.Ecras.ResolucaoEcras.designacao == "1280x720 (HD)")
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);
    if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"1280x720 (HD)" && caso.Ecras.ResolucaoEcras.designacao == "2560x1440 (2K)")
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);

        if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"1280x720 (HD)" && caso.Ecras.ResolucaoEcras.designacao == "3840x2160 (4K)")
            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.12);
            if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"3840x2160 (4K)" && caso.Ecras.ResolucaoEcras.designacao == "1280x720 (HD)")
                DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.12);
                //-----
                if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"2560x1440 (2K)" && caso.Ecras.ResolucaoEcras.designacao == "1920x1080 (FULL HD)")
                    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.06);
                    if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"1920x1080 (FULL HD)" && caso.Ecras.ResolucaoEcras.designacao == "2560x1440 (2K)")
                        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.06);

                        if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"3840x2160 (4K)" && caso.Ecras.ResolucaoEcras.designacao == "1920x1080 (FULL HD)")
                            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);
                            if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcras == e.idResolucaoEcras).designacao ==
"1920x1080 (FULL HD)" && caso.Ecras.ResolucaoEcras.designacao == "3840x2160 (4K)")
                                DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);

```

```
//-----

if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação == "2560x1440 (2K)")

    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.04);

    if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra == e.idResoluçãoEcra).designação ==
"2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação == "3840x2160 (4K)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.04);

}

if (e.tamanho > caso.Ecra.tamanho)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeTamanhoEcra(e.tamanho, caso.Ecra.tamanho)) * 0.025);
}

if (e.tamanho < caso.Ecra.tamanho)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeTamanhoEcra(e.tamanho, caso.Ecra.tamanho)) * 0.025);
}

if (p.velocidadeProcessador > caso.Processador.velocidadeProcessador)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeVelocidadeProcessador(p.velocidadeProcessador, caso.Processador.velocidadeProcessador)) * 0.05);
}

if (p.velocidadeProcessador < caso.Processador.velocidadeProcessador)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeVelocidadeProcessador(p.velocidadeProcessador, caso.Processador.velocidadeProcessador)) * 0.05);
}
```

```

    }

    if (p.nucleosProcessador > caso.Processador.nucleosProcessador)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeNucleosProcessador(p.nucleosProcessador, caso.Processador.nucleosProcessador)) * 0.025);
    }

    if (p.nucleosProcessador < caso.Processador.nucleosProcessador)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeNucleosProcessador(p.nucleosProcessador, caso.Processador.nucleosProcessador)) * 0.025);
    }

    if (c.resolucaoFrontal > caso.Camera.resolucaoFrontal)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeResolucaoCameraFrontal(c.resolucaoFrontal, caso.Camera.resolucaoFrontal)) * 0.05);
    }

    if (c.resolucaoFrontal < caso.Camera.resolucaoFrontal)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeResolucaoCameraFrontal(c.resolucaoFrontal, caso.Camera.resolucaoFrontal)) * 0.05);
    }

    if (c.resolucaoTraseira > caso.Camera.resolucaoTraseira)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimilidadeResolucaoCameraTraseira(c.resolucaoTraseira, caso.Camera.resolucaoTraseira)) * 0.05);
    }

    if (c.resolucaoTraseira < caso.Camera.resolucaoTraseira)

```

```

{
    DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * (1 -
SimilidadeResolucaoCameraTraseira(c.resolucaoTraseira, caso.Camera.resolucaoTraseira)) * 0.05);
}

if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação != caso.Estado.designação)
{
    int e1 = 0, e2 = 0;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Péssimo") e1 = 0;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Mau") e1 = 1;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Intermédio") e1 = 2;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Bom") e1 = 3;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Muito Bom") e1 = 4;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Como Novo") e1 = 5;

    if (caso.Estado.designação == "Péssimo") e2 = 0;
    if (caso.Estado.designação == "Mau") e2 = 1;
    if (caso.Estado.designação == "Intermédio") e2 = 2;
    if (caso.Estado.designação == "Bom") e2 = 3;
    if (caso.Estado.designação == "Muito Bom") e2 = 4;
    if (caso.Estado.designação == "Como Novo") e2 = 5;

    int resultado = (5 - Math.Abs(e1 - e2)) / 5;
    if (e1 > e2) //impossível ser igual, verificado antes
        DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * ((1 - resultado) * 0.1));
    else
        DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * ((1 - resultado) * 0.1));
}

if (tm.idade > caso.idade)
{
    DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * (1 -
SimilidadeIdade(tm.idade, caso.idade)) * 0.1);
}

```

```
if (tm.idade < caso.idade)
{
    DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * (1 -
SimiliridadeIdade(tm.idade, caso.idade)) * 0.1);
}
DataBase.SubmitChanges();
}
```

3.4. Funções de similaridade

De seguida, são apresentadas as funções de similaridade e uma explicação individual para cada função, cada característica é analisada por uma função específica de modo a ser mais específica quanto ao grau de similaridade.

```
public double SimiliridadeMarca(string marca1, string marca2)
{
    if (marca1 == marca2) return 1;
    else if (marca1 == "Apple" && marca2 == "Samsung") return 0.8; //similaridade em termos de preço
    else if (marca1 == "Samsung" && marca2 == "Apple") return 0.8;
    else if (marca1 == "LG" && marca2 == "Samsung") return 0.5;
    else if (marca1 == "Samsung" && marca2 == "LG") return 0.5;
    else if (marca1 == "LG" && marca2 == "Apple") return 0.4;
    else if (marca1 == "Apple" && marca2 == "LG") return 0.4;
    else if (marca1 == "Outras" && marca2 == "LG") return 0.5;
    else if (marca1 == "LG" && marca2 == "Outras") return 0.5;
    else if (marca1 == "Outras" && marca2 == "Samsung") return 0.3;
    else if (marca1 == "Samsung" && marca2 == "Outras") return 0.3;
    else if (marca1 == "Outras" && marca2 == "Apple") return 0.1;
    else if (marca1 == "Apple" && marca2 == "Outras") return 0.1;
    else return 0; //nunca poderá acontecer ser 0 porque a view não permite outras opções além das de cima.
}
```

São comparadas a totalidade de opções de marcas disponíveis, e retornado um valor, escolhido após alguns testes e principalmente tendo em conta a notoriedade das marcas e o valor que pedem pelos seus modelos. Usou-se *if's* para um processamento mais rápido.

```
public double SimiliridadeRam(double ram1, double ram2)
{
    if (ram1 >= 10 && ram2 >= 10) return 1; //acima de 10 a ram é totalmente indiferente para um telemovel
    if (Math.Abs(ram1 - ram2) > 10) return 0; //nao teem semelhança nenhuma
    else
    {
        if (ram1 <= 2 && ram2 <= 2) //objetivo: acentuar a diferença para ram's com valores abaixo de 2
            return (4 - Math.Abs(ram1 - ram2)) / 4; //varia de 0.5 a 1
        //exemplo: uma diferença de 1gb para 2gb é mais substancial do que 8gb para 9gb
        else
            return (10 - Math.Abs(ram1 - ram2)) / 10;
    }
}
```

A função de similaridade da memória RAM é apresentada acima, verificou-se que não existem memórias RAM com mais de 10 GBs para telemóveis, além de ser também inútil para o desempenho de um telemóvel e portanto qualquer memória RAM com 10 GBs tem o mesmo valor de uma com mais que 10 GBs, portanto a similaridade nesse caso é sempre 1.

Se as memórias RAM têm uma distancia de mais de 10 GBs, não são comparáveis, têm similaridade 0.

Caso contrário, é aplicada uma função para as memórias RAM com menos ou igual a 2 GBs e outra para as restantes memórias RAM (2 a 8 GBs). Verificou-se que fazendo apenas uma função, a similaridade entre uma memória RAM, exemplificando, de 1 GBs para uma memória RAM de 2 GBs seria igual à similaridade entre as memórias RAM de 9 GBs e 10 GBs porque a diferença em ambos os casos é igual. Isto não é correto porque no primeiro caso estamos a falar de 2 memórias bem diferentes e no segundo de memórias praticamente iguais. Portanto pretendeu-se acentuar a diferença para memórias com menos de 2 GBs, para estas a similaridade variará de 0.5 a 1 em qualquer dos casos com a nova função, o que com funções testadas anteriormente variava entre 0.8 e 1. Para as restantes memórias é utilizada a função linear para o valor ideal 10, também normalizada de 0 a 1.


```
public double SimilidadeMemoriaInterna(int mInterna1, int mInterna2)
{
    if (mInterna1 >= 256 && mInterna2 >= 256) return 1; //acima de 256 a memoria nao tem importancia,
    sao iguais em termos de preço
    if (Math.Abs(mInterna1 - mInterna2) > 256) return 0; //nao teem semelhança nenhuma
    else
    {
        if (mInterna1 <= 5 && mInterna2 <= 5) //objetivo: acentuar a diferença para memorias baixas
            return (10 - Math.Abs(mInterna1 - mInterna2)) / 10; //varia de 0.5 a 1
        //exemplo: uma diferença de 5gb para 15gb é mais substancial do que 50gb para 60gb
        else
            return (256 - Math.Abs(mInterna1 - mInterna2)) / 256;
    }
}
```

A função de similaridade da memória é apresentada acima, verificou-se que não existem memórias com mais de 256 GBs para telemóveis, além de o espaço ser já enorme para qualquer utilizador de telemóvel, portanto considera-se que uma memória de 256 GBs tem o mesmo valor de memórias com mais que de 256 GBs, portanto a similaridade nesse caso é sempre 1.

Se as memórias têm uma distancia de mais de 256 GBs, não são comparáveis, têm similaridade 0.

Caso contrário, é aplicada uma função para as memórias com menos de 5 GBs e outra para as restantes memórias RAM (5 a 256 GBs). O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimiliridademAhBateria(int mAhBateria1, int mAhBateria2)
{
    if (mAhBateria1 >= 5000 && mAhBateria2 >= 5000) return 1; //não existem baterias com mais mAh
    if (Math.Abs(mAhBateria1 - mAhBateria2) > 5000) return 0;
    else
    {
        if (mAhBateria1 <= 1500 && mAhBateria2 <= 1500)
            return (3000 - Math.Abs(mAhBateria1 - mAhBateria2)) / 3000; //varia de 0.5 a 1
        else
            return (5000 - Math.Abs(mAhBateria1 - mAhBateria2)) / 5000;
    }
}
```

A função de similaridade da potência da bateria é apresentada acima, verificou-se que não existem baterias com mais de 5000 mAh para telemóveis, portanto considera-se que uma bateria com 5000 mAh tem o mesmo valor de baterias com mais que de 5000 mAh, portanto a similaridade nesse caso é sempre 1. Se as baterias têm uma distancia de mais de 5000 mAh, não são comparáveis, têm similaridade 0.

Caso contrário, é aplicada uma função para as memórias com menos de 1500 mAh e outra para as restantes baterias (1500 a 5000 mAh). O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimiliridadeResoluçãoEcra(string res1, string res2)
{
    //"640x480 (VGA)"
    //"1280x720 (HD)"
    //"1920x1080 (FULL HD)"
    //"2560x1440 (2K)"
    //"3840x2160 (4K)"
    if (res1 == res2) return 1;

    if (res1 == "640x480 (VGA)" && res2 == "1280x720 (HD)") return 0.5;
    if (res1 == "1280x720 (HD)" && res2 == "640x480 (VGA)") return 0.5;

    if (res1 == "640x480 (VGA)" && res2 == "1920x1080 (FULL HD)") return 0.4;
```

```

if (res1 == "1920x1080 (FULL HD)" && res2 == "640x480 (VGA)") return 0.4;

if (res1 == "640x480 (VGA)" && res2 == "2560x1440 (2K)") return 0.2;
if (res1 == "2560x1440 (2K)" && res2 == "640x480 (VGA)") return 0.2;

if (res1 == "640x480 (VGA)" && res2 == "3840x2160 (4K)") return 0;
if (res1 == "3840x2160 (4K)" && res2 == "640x480 (VGA)") return 0;
//-----
if (res1 == "1920x1080 (FULL HD)" && res2 == "1280x720 (HD)") return 0.7;
if (res1 == "1280x720 (HD)" && res2 == "1920x1080 (FULL HD)") return 0.7;

if (res1 == "2560x1440 (2K)" && res2 == "1280x720 (HD)") return 0.5;
if (res1 == "1280x720 (HD)" && res2 == "2560x1440 (2K)") return 0.5;

if (res1 == "1280x720 (HD)" && res2 == "3840x2160 (4K)") return 0.2;
if (res1 == "3840x2160 (4K)" && res2 == "1280x720 (HD)") return 0.2;
//-----
if (res1 == "2560x1440 (2K)" && res2 == "1920x1080 (FULL HD)") return 0.6;
if (res1 == "1920x1080 (FULL HD)" && res2 == "2560x1440 (2K)") return 0.6;

if (res1 == "3840x2160 (4K)" && res2 == "1920x1080 (FULL HD)") return 0.4;
if (res1 == "1920x1080 (FULL HD)" && res2 == "3840x2160 (4K)") return 0.4;
//-----
if (res1 == "3840x2160 (4K)" && res2 == "2560x1440 (2K)") return 0.5;
if (res1 == "2560x1440 (2K)" && res2 == "3840x2160 (4K)") return 0.5;
else return 0;
}

```

São comparadas a totalidade de opções de resoluções disponíveis, e retornado um valor, escolhido após alguns testes e principalmente tendo em conta o valor das resoluções e o valor que pedem pelos seus modelos. Usou-se *if's* para um processamento mais rápido.

```
public double SimiliridadeTamanhoEcra(double tamanho1, double tamanho2)
{
    //tamanho ideal de um ecra para telemovel segundo a maioria das sondagens: 5 polegadas
    if (tamanho1 > 10 && tamanho2 > 10) return 1;
    if (Math.Abs(tamanho1 - tamanho2) < 5) return 5 - (Math.Abs(tamanho1 - tamanho2)) / 5;
    else
    {
        if (Math.Abs(tamanho1 - tamanho2) == 5) return 1;
        else
            return (10 - Math.Abs(tamanho1 - tamanho2)) / 5;
    }
}
```

A função de similaridade do tamanho de ecrã é apresentada acima, verificou-se, a partir de pesquisas, que as sondagens encontradas constatavam que o tamanho de ecrã preferido dos utilizadores é 5 polegadas, logo um telemóvel perderá valor caso tenha tamanho maior ou menor.

Se os tamanhos são ambos maiores que 10, o valor retornado é 1, porque qualquer tamanho maior que 10 não tem valor para um utilizador de telemóveis, além de nem existirem telemóveis com tal tamanho de ecrã.

Casos os telemóveis tenham tamanhos inferiores a 5, é aplicada uma função de similaridade linear, caso tenham tamanho 5, é retornado 1 e caso tenham tamanho maior que 5 é aplicada outra função de similaridade linear, variando de 0 a 1.

```
public double SimiliridadeVelocidadeProcessador(double vel1, double vel2)
{
    //acima de 5GHz é indiferente
    if (vel1 >= 5 && vel2 >= 5) return 1;
    if (Math.Abs(vel1 - vel2) > 5) return 0;
    else
    {
        if (vel1 <= 1500 && vel2 <= 1.5)
            return (3 - Math.Abs(vel1 - vel2)) / 3; //varia de 0.5 a 1
        else
            return (5 - Math.Abs(vel1 - vel2)) / 5000;
    }
}
```

A função de similaridade da velocidade do processador é apresentada acima, verificou-se que não existem velocidades de processador com mais de 5 GHz para telemóveis, portanto considera-se que uma velocidade de 5 GHz tem o mesmo valor de velocidades com mais que de 5 GHz, portanto a similaridade nesse caso é sempre 1. Se as velocidades têm uma distancia de mais de 5 GHz, não são comparáveis, têm similaridade 0.

Caso contrário, é aplicada uma função para as velocidades com menos de 3 GHz e outra para as restantes velocidades (3 a 5 GHz). O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimilidadeNucleosProcessador(int n1, int n2)
{ //é inutil ter mais que 16 nucleos
    if (n1 >= 16 && n2 >= 16) return 1;
    if (Math.Abs(n1 - n2) > 16) return 0;
    else
    {
        if (n1 <= 2 && n2 <= 2)
            return (4 - Math.Abs(n1 - n2)) / 4; //varia de 0.5 a 1
        else
            return (16 - Math.Abs(n1 - n2)) / 16;
    }
}
```

A função de similaridade dos núcleos do processador é apresentada acima, verificou-se que não existem telemóveis com mais de 16 núcleos, portanto considera-se mais de 16 núcleos é indiferente e retorna-se 1.

Se os telemóveis têm diferença e 16 núcleos, não são comparáveis, têm similaridade 0.

Caso contrário, é aplicada uma função para telemóveis com 2 ou menos núcleos e outra para os restantes. O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimiliridadeResolucaoCameraFrontal(double cFrontal1, double cFrontal2)
{
    //no mercado nao existem camaras frontais com mais de 16 megapixels
    if (cFrontal1 >= 16 && cFrontal2 >= 16) return 1;
    if (Math.Abs(cFrontal1 - cFrontal2) > 16) return 0;
    else
    {
        if (cFrontal1 <= 4 && cFrontal2 <= 4)
            return (8 - Math.Abs(cFrontal1 - cFrontal2)) / 8; //varia de 0.5 a 1
        else
            return (16 - Math.Abs(cFrontal1 - cFrontal2)) / 16;
    }
}
```

A função de similaridade da camara frontal é apresentada acima, verificou-se que não existem camaras frontais com mais de 16 *megapixels* para telemóveis, portanto considera-se que camaras acima disso são idênticas e retorna-se 1.

Caso tenham distância 16 retorna-se 0.

Caso contrário, é aplicada uma função para as camaras com menos de 4 *megapixels* e outra para *megapixels* entre 4 e 16. O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimiliridadeResolucaoCameraTraseira(double cTraseira1, double cTraseira2)
{
    if (cTraseira1 >= 20 && cTraseira2 >= 20) return 1;
    if (Math.Abs(cTraseira1 - cTraseira2) > 20) return 0;
    else
    {
        if (cTraseira1 <= 8 && cTraseira2 <= 8)
            return (16 - Math.Abs(cTraseira1 - cTraseira2)) / 16; //varia de 0.5 a 1
        else
            return (20 - Math.Abs(cTraseira1 - cTraseira2)) / 20;
    }
}
```

A função de similaridade da camara traseira é apresentada acima, verificou-se que não existem camaras traseiras com mais de 20 *megapixels* para telemóveis, portanto considera-se que camaras acima disso são idênticas e retorna-se 1.

Caso tenham distância 20 retorna-se 0.

Caso contrário, é aplicada uma função para as camaras com menos de 8 *megapixels* e outra para *megapixels* entre 8 e 20. O motivo é exatamente o descrito acima para as memórias RAM.

```
public double SimilidadeEstado(string estado1, string estado2)
{
    int e1 = 0, e2 = 0;
    if (estado1 == "Péssimo") e1 = 0;
    if (estado1 == "Mau") e1 = 1;
    if (estado1 == "Intermédio") e1 = 2;
    if (estado1 == "Bom") e1 = 3;
    if (estado1 == "Muito Bom") e1 = 4;
    if (estado1 == "Como Novo") e1 = 5;

    if (estado2 == "Péssimo") e2 = 0;
    if (estado2 == "Mau") e2 = 1;
    if (estado2 == "Intermédio") e2 = 2;
    if (estado2 == "Bom") e2 = 3;
    if (estado2 == "Muito Bom") e2 = 4;
    if (estado2 == "Como Novo") e2 = 5;

    return (5 - Math.Abs(e1 - e2)) / 5;
}
```

Para os estados utilizou-se uma função linear, primeiro traduzindo os estados para inteiros e depois fazendo a diferença entre os mesmos, normalizada de 0 a 1.

```
public double SimiliridadeIdade(int idade1, int idade2)
{
    //dada em meses
    if (idade1 >= 48 && idade2 >= 48) return 1; //acima de 4 anos é igual
    if (Math.Abs(idade1 - idade2) > 48) return 0;
    else
    {
        if (idade1 <= 6 && idade2 <= 6) //a diferença é mais relevante
            return (12 - Math.Abs(idade1 - idade2)) / 12; //varia de 0.5 a 1
        else
            return (48 - Math.Abs(idade1 - idade2)) / 48;
    }
}
```

A função de similaridade da idade é apresentada acima, verifica-se que telemóveis com mais de 4 anos (48 meses) são telemóveis com pouco valor e retorna-se 1 neste caso, pois é idêntico terem 4 ou mais anos. Caso os telemóveis tenham diferença de 48 meses são completamente diferentes e retorna-se 0.

Caso contrário, é aplicada uma função para idade de 6 ou menos meses e outra para idades entre 6 e 48 meses. O motivo é exatamente o descrito acima para as memórias RAM.

4. Implementação

Para a implementação do programa utilizou-se uma base de dados definida em *Microsoft SQL Server*, para a interface gráfica utilizou-se *HTML* e para a codificação do algoritmo utilizou-se *ASP.NET*, utilizando o compilador do *Microsoft Visual Studio Enterprise 2015*.

4.1. Diagrama SQL

De seguida apresenta-se o diagrama SQL referente à base de dados implementada:

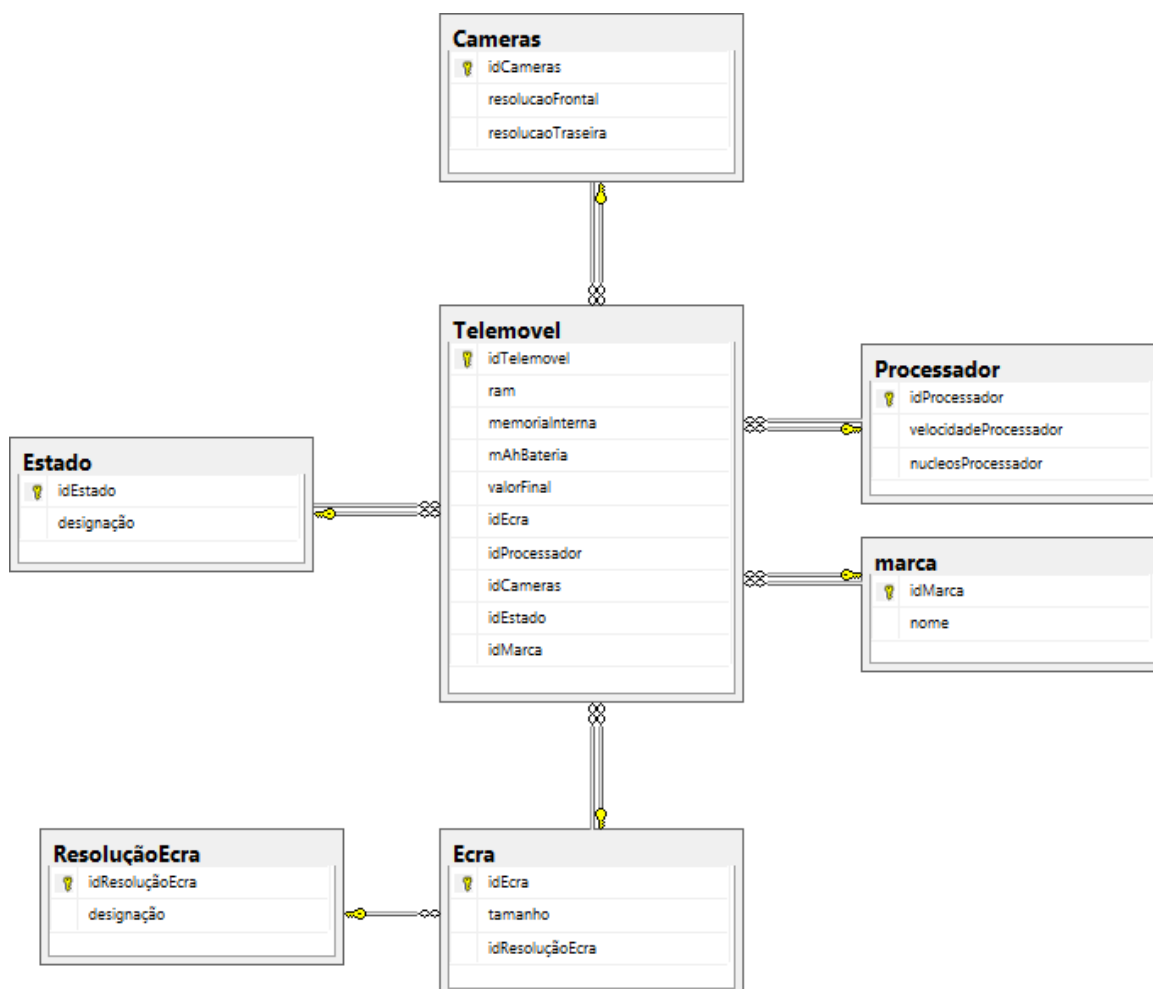
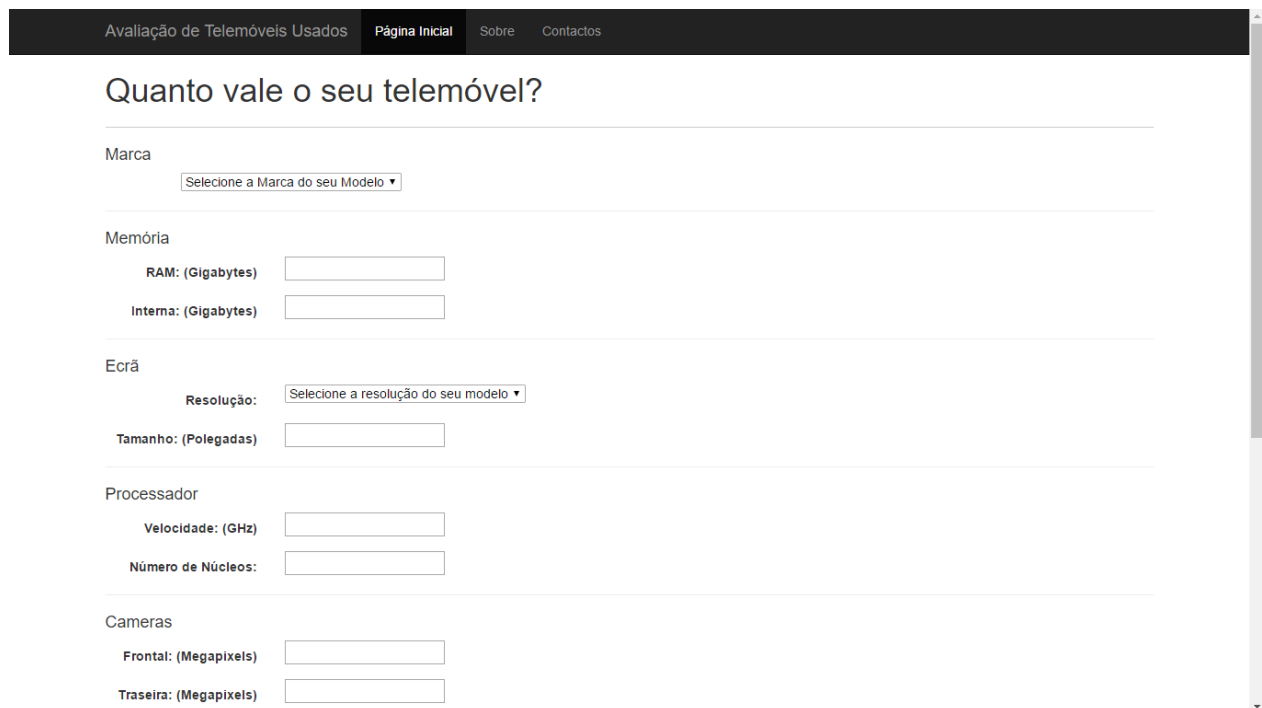


Figura 2: Diagrama da Base de Dados

4.2. Views

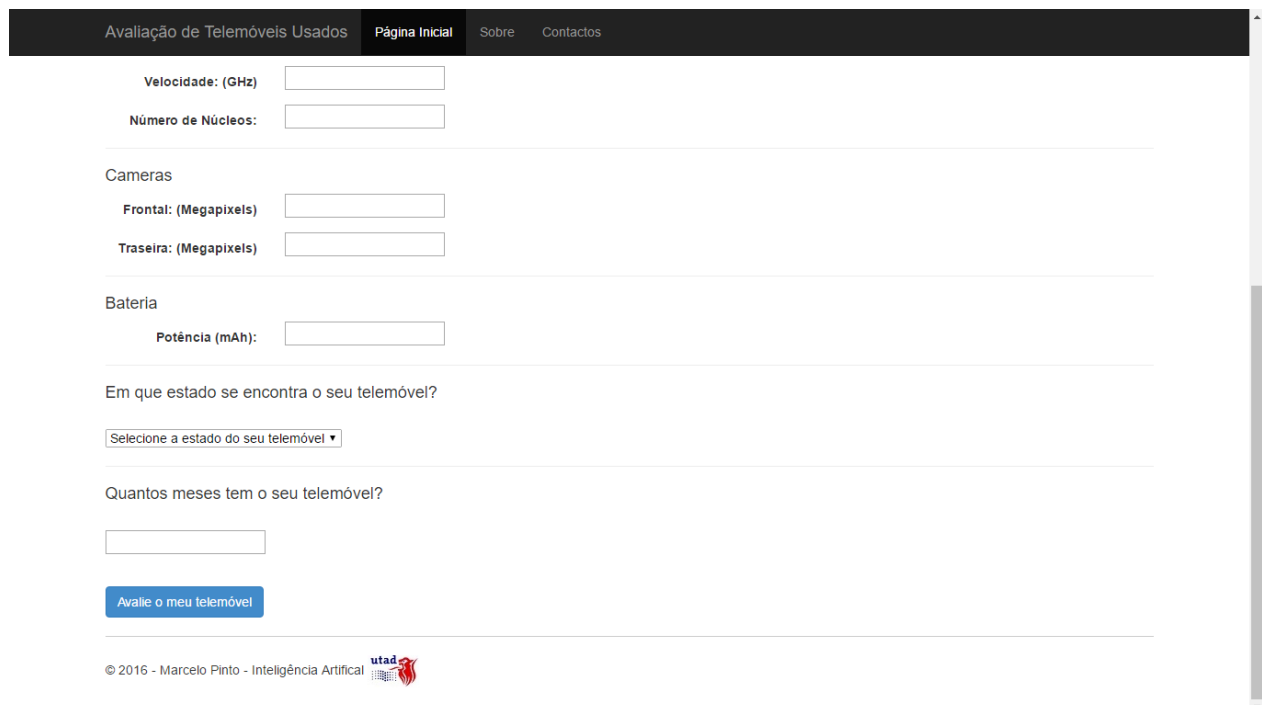
De seguida apresenta-se a *view* de avaliação:



The screenshot shows a web form titled "Quanto vale o seu telemóvel?". The form is divided into several sections, each with a label and input fields:

- Marca:** A dropdown menu with the text "Selecione a Marca do seu Modelo".
- Memória:** Two input fields: "RAM: (Gigabytes)" and "Interna: (Gigabytes)".
- Ecrã:** Two input fields: "Resolução:" (with a dropdown menu "Selecione a resolução do seu modelo") and "Tamanho: (Polegadas)".
- Processador:** Two input fields: "Velocidade: (GHz)" and "Número de Núcleos".
- Cameras:** Two input fields: "Frontal: (Megapixels)" and "Traseira: (Megapixels)".

Figura 3: View Avaliação



The screenshot shows the second part of the evaluation form. It continues with input fields and a final button:

- Velocidade: (GHz):** Input field.
- Número de Núcleos:** Input field.
- Cameras:** Two input fields: "Frontal: (Megapixels)" and "Traseira: (Megapixels)".
- Bateria:** Input field: "Potência (mAh):".
- Em que estado se encontra o seu telemóvel?:** A dropdown menu with the text "Selecione a estado do seu telemóvel".
- Quantos meses tem o seu telemóvel?:** Input field.
- Button:** A blue button labeled "Avalie o meu telemóvel".
- Footer:** Copyright notice: "© 2016 - Marcelo Pinto - Inteligência Artificial" and the utad logo.

Figura 4: View Avaliação - Parte 2

Após o clique no botão “Avalie o meu telemóvel”, o processamento dos algoritmos de CBR são executados e o resultado aparecerá:

[Avaliação de Telemóveis Usados](#) [Página Inicial](#) [Sobre](#) [Contactos](#)

Avaliação: 98 €

Telemovel:

| | |
|----------------------|---------------|
| marca | LG |
| Bateria (mAh) | 2100 |
| RAM (GB) | 2 |
| Memória interna (GB) | 8 |
| Tamanho Ecrã | 5,5 |
| Resolução Ecrã | 1280x720 (HD) |
| Velocidade (GHz) | 1,4 |
| Núcleos Processador | 2 |
| Camera Frontal (MP) | 5 |
| Camera traseira (MP) | 8 |
| Estado | Bom |
| Idade (Meses) | 10 |

[Avaliar outro telemóvel.](#)


© 2016 - Marcelo Pinto - Inteligência Artificial 

Figura 5: View Valor da Avaliação

Ao administrador, será disponibilizada o acesso à lista de casos assim como e inserção, edição e eliminação de casos, as *views* são apresentadas de seguida:

Avaliação de Telemóveis Usados

Página Inicial

Sobre

Contactos

Administrador

[Novo Caso](#)

Base de Casos

| Marca | Bateria (mAh) | RAM (GB) | Memória Interna (GB) | Tamanho Ecrã (Polegadas) | Resolução Ecrã | Velocidade Processador (GHz) | Núcleos Processador | Camera Frontal (MP) | Camera traseira (MP) | Estado | Idade (Meses) | Valor Final (Euros) | |
|---------|---------------|----------|----------------------|--------------------------|---------------------|------------------------------|---------------------|---------------------|----------------------|------------|---------------|---------------------|--|
| Apple | 3000 | 4 | 64 | 6 | 3840x2160 (4K) | 3 | 8 | 12 | 16 | Como Novo | 2 | 450 | Editar Eliminar |
| Outras | 2100 | 1,3 | 8 | 4,7 | 1280x720 (HD) | 1,4 | 4 | 1,3 | 8 | Bom | 26 | 120 | Editar Eliminar |
| Samsung | 2100 | 2 | 16 | 5,5 | 1920x1080 (FULL HD) | 2,7 | 8 | 4 | 8 | Muito Bom | 6 | 200 | Editar Eliminar |
| LG | 2500 | 2 | 8 | 5 | 1280x720 (HD) | 2,5 | 4 | 2 | 8 | Muito Bom | 6 | 140 | Editar Eliminar |
| LG | 2100 | 2 | 8 | 5 | 1280x720 (HD) | 1 | 4 | 2 | 8 | Muito Bom | 7 | 100 | Editar Eliminar |
| Samsung | 2000 | 2 | 8 | 2 | 2560x1440 (2K) | 2 | 4 | 2 | 8 | Muito Bom | 6 | 200 | Editar Eliminar |
| Outras | 2300 | 2 | 6 | 5 | 2560x1440 (2K) | 1,3 | 4 | 4 | 8 | Intermédio | 12 | 90 | Editar Eliminar |
| LG | 2150 | 2 | 8 | 4 | 2560x1440 (2K) | 2 | 2 | 5 | 12 | Muito Bom | 2 | 150 | Editar Eliminar |
| Samsung | 2200 | 2 | 16 | 4 | 3840x2160 (4K) | 2 | 4 | 4 | 12 | Intermédio | 6 | 265 | Editar Eliminar |

Figura 6: View da Base de Casos

| Avaliação de Telemóveis Usados | | | | | | | | | | | | |
|---|--|--|--|--|--|--|--|--|--|--|--|--|
| Página Inicial | | | | | | | | | | | | |
| Sobre | | | | | | | | | | | | |
| Contactos | | | | | | | | | | | | |
| Administrador | | | | | | | | | | | | |
| Base de Casos | | | | | | | | | | | | |
| Adicione um Caso: | | | | | | | | | | | | |
| Marca <div> Seleccione a Marca do Modelo ▼ </div> | | | | | | | | | | | | |
| Memória <div> RAM: (Gigabytes) <input type="text"/> Interna: (Gigabytes) <input type="text"/> </div> | | | | | | | | | | | | |
| Ecrã <div> Resolução: Seleccione a resolução do modelo ▼ Tamanho: (Polegadas) <input type="text"/> </div> | | | | | | | | | | | | |
| Processador <div> Velocidade: (GHz) <input type="text"/> Número de Núcleos: <input type="text"/> </div> | | | | | | | | | | | | |
| Cameras <div> Frontal: (Megapixels) <input type="text"/> </div> | | | | | | | | | | | | |

Figura 7: View Adicionar Caso

Avaliação de Telemóveis Usados
Página Inicial
Sobre
Contactos

Administrador

Editar Caso:

Marca

Apple

Memória

RAM: (Gigabytes) 4

Interna: (Gigabytes) 64

Ecrã

Resolução: 3840x2160 (4K)

Tamanho: (Polegadas) 6

Processador

Velocidade: (GHz) 3

Número de Núcleos: 8

Cameras

Frontal: (Megapixels) 12

Traseira: (Megapixels) 16

Figura 8: View Editar Caso

Avaliação de Telemóveis Usados
Página Inicial
Sobre
Contactos

Administrador

Tem a certeza que quer apagar este telemóvel da Base de Casos?

| | |
|----------------------|----------------|
| marca | Apple |
| Bateria (mAh) | 3000 |
| RAM (GB) | 4 |
| Memória Interna (GB) | 64 |
| Tamanho | 6 |
| Resolução Ecrã | 3840x2160 (4K) |
| Velocidade (GHz) | 3 |
| Núcleos Processador | 8 |
| Camera Frontal (MP) | 12 |
| Camera traseira (MP) | 16 |
| Estado | Como Novo |
| Idade (Meses) | 2 |
| Valor Final (Euros) | 450 |

[Base de Casos](#)

© 2016 - Marcelo Pinto - Inteligência Artificial 

Figura 9: View Confirmação de eliminação de Caso

O utilizador ainda terá acesso a mais 2 *views*, uma acerca do site e outra para contactos.

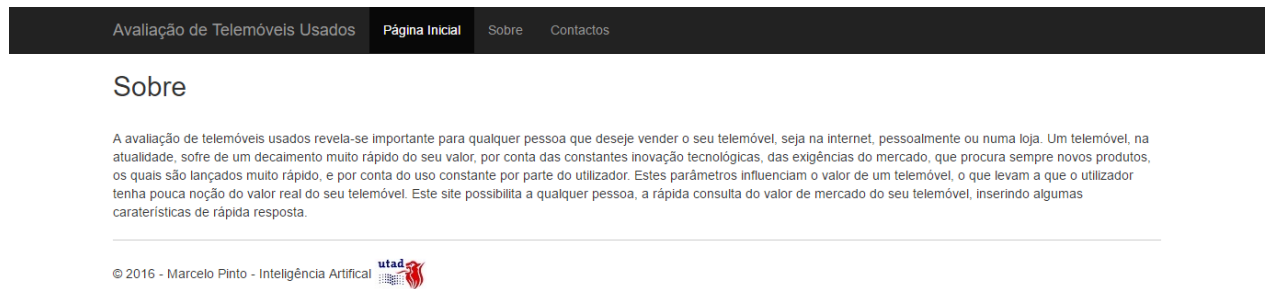


Figura 10: View Sobre

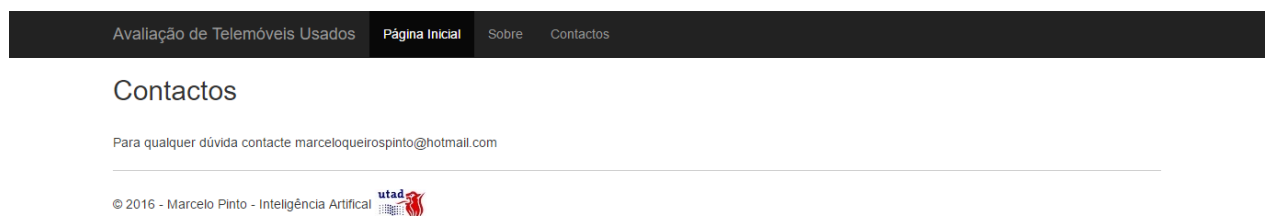


Figura 11: View Contactos

5. Conclusão

Concluindo, constata-se que todos os objetivos foram largamente atingidos, e mesmo os algoritmos geram valores de avaliação totalmente dentro dos valores de mercado praticados.

Verifica-se que este tipo de técnica é passível de ser aplicado às mais diversas áreas de estudo, para resolver problemas na análise de dados, na previsão de soluções, entre outros. Verifica-se que existem muitas variações e condicionantes no que toca à implementação deste tipo de soluções, e, portanto, deve-se ter cuidado na escolha de pesos, nas funções de similaridade e na fase revisão, de modo a estes elementos coincidirem com a realidade do mercado. Deste modo, em casos reais, é necessário a colaboração de um especialista acerca da área.

6. Bibliografia/Referências

Leake, D. B. (1996). *CBR in Context: The Present and Future* (AAAI Press/MIT Press). Consultado a 28 de Novembro em <http://www.cs.indiana.edu/pub/leake/p-96-01.pdf>

Castoldi, A. & Santos, M.O. (2002). *Raciocínio Baseado em Casos*. Consultado em 28 de Novembro em <http://www.inf.ufsc.br/~barreto/trabaluno/ia20022augmarc.pdf>

Kolodner, J.L. (1993). *Case-Based Learning*. Consultado a 29 de Novembro em <http://www.google.pt/books?id=CC2-3VA8UfwC&printsec=frontcover&hl=pt-PT#v=onepage&q&f=false>

Aamodt, A. & Plaza, E. (1994). *Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches*. Consultado a 29 de Novembro em: http://www.idi.ntnu.no/emner/tdd4173/papers/Aamodt_1994_Case.pdf

Abel, M. (1996). *Um estudo sobre Raciocínio Baseado em Casos*. Consultado a 29 de Novembro em: <http://www.inf.ufrgs.br/bdi/wp-content/uploads/CBR-TI60.pdf>

Castoldi, A. & Santos, M.O. (2002). *Raciocínio Baseado em Casos*. Consultado a 29 de Novembro em: <http://www.inf.ufsc.br/~barreto/trabaluno/ia20022augmarc.pdf>

Olsson, T. & Funk P. *Case-based reasoning combined with statistics for diagnostics and prognosis*. Consultado a 29 de Novembro em <http://iopscience.iop.org/article/10.1088/1742-6596/364/1/012061/pdf>

Mulyana S. & Hartati S. & Wardoyo R. & Winarko E. *Case-Based Reasoning for Selecting Study Program in Senior High School*. Consultado a 29 de Novembro em http://thesai.org/Downloads/Volume6No4/Paper_18-Case-Based Reasoning for Selecting Study Program in Senior High School.pdf

Kerstin Bach & Klaus-Dieter Althoff & Julian Satzky & Julian Kroehl. *CookIIS Mobile: A Case-Based Reasoning Recipe Customizer for Android Phones*. Consultado a 29 de Novembro em https://www.google.pt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjf7Jvgz9PQAhUG1RoKHeZOBACQFggfMAA&url=https%3A%2F%2Fwww.dfki.de%2Fweb%2Fforschung%2Fkm%2Fpublikationen%2FrenameFileForDownload%3Ffilename%3D2012-cookiis-ukcbr.pdf%26file_id%3Duploads_1867&usg=AFQjCNE3k6sohKIDy1Q49mavr84OBNcpSA&sig2=GBclMbTkQw8VTb7VPaTk_A

Pei-Chann & Chang & Jyun-Jie Lin & Wei-Yuan Dzan. *Forecasting of manufacturing cost in mobile phone products by case-based reasoning and artificial neural network models*. Consultado a 29 de Novembro em <http://link.springer.com/article/10.1007/s10845-010-0390-7>

7. Anexos

De seguida apresenta-se o código Asp.Net e o código da base de dados. O código HTML não será apresentado porque possui grande dimensão, mas poderá ser consultado nos ficheiros que são enviados com o trabalho.”.

7.1. Asp.Net

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Web;
```

```
namespace Modulo2IA_CBR_Telemoveis.Models
```

```
{
```

```
    public class Data
```

```
    {
```

```
        public DataClassesCBRDataContext DataBase { get; set; }
```

```
        public Telemovel tm { get; set; }
```

```
        public Camera c { get; set; }
```

```
        public Ecra e { get; set; }
```

```
        public Processador p { get; set; }
```

```
        public Data()
```

```
        {
```

```
            tm = new Telemovel();
```

```
            e = new Ecra();
```

```
            c = new Camera();
```

```
            p = new Processador();
```

```
            DataBase = new DataClassesCBRDataContext();
```

```
        }
```

```
        public void SetTelemovel(int _idMarca, int _idResolução, int _idEstado, double _ram, int  
_memoriaInterna, int _mAhBateria,
```

```
double _tamanhoEcra, double _velocidadeProcessador, int _nucleosProcessador, double
_resolucaoCameraFrontal,
double _resolucaoCameraTraseira, int _idade)
{
    tm.idMarca = _idMarca; //id para ligação à classe já existente
    e.idResoluçãoEcra = _idResolução; //id para ligação à classe já existente
    tm.idEstado = _idEstado; //id para ligação à classe já existente
    tm.ram = _ram;          //quantidade de memoria ram (é dada em quantidades que podem
ser floats)
    tm.memoriaInterna = _memoriaInterna; //quantidade de memoria interna
    tm.mAhBateria = _mAhBateria; //mAh da bateria
    e.tamanho = _tamanhoEcra; //em polegadas
    p.velocidadeProcessador = _velocidadeProcessador;
    p.nucleosProcessador = _nucleosProcessador; //numero de nucleos do processador
    c.resolucaoFrontal = _resolucaoCameraFrontal; //resolução da camera frontal
    c.resolucaoTraseira = _resolucaoCameraTraseira; //resolução da camera traseira
    tm.idade = _idade; //dada em meses
}

public void AdicionarCaso()
{
    submeterNaDB();
    tm = new Telemovel();
    e = new Ecra();
    c = new Camera();
    p = new Processador();
}

public void EditarCaso(int id, int _idMarca, int _idResolução, int _idEstado, double _ram, int
_memoriaInterna, int _mAhBateria,
double _tamanhoEcra, double _velocidadeProcessador, int _nucleosProcessador, double
_resolucaoCameraFrontal,
```

```

double _resolucaoCameraTraseira, int _idade, int _valor)
{
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).idMarca = _idMarca;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).Ecra.idResoluçãoEcra =
_idResolução;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).idEstado = _idEstado;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).ram = _ram;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).memoriaInterna =
_memoriaInterna;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).mAhBateria = _mAhBateria;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).Ecra.tamanho = _tamanhoEcra;
    DataBase.Telemoveis.Single(x => x.idTelemovel ==
id).Processador.velocidadeProcessador = _velocidadeProcessador;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).Processador.nucleosProcessador
= _nucleosProcessador;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).Camera.resolucaoFrontal =
_resolucaoCameraFrontal;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).Camera.resolucaoTraseira =
_resolucaoCameraTraseira;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).idade = _idade;
    DataBase.Telemoveis.Single(x => x.idTelemovel == id).valorFinal = _valor;
    DataBase.SubmitChanges();
}

public void submeterNaDB()
{
    DataBase.Cameras.InsertOnSubmit(c);
    DataBase.Ecras.InsertOnSubmit(e);
    DataBase.Processadors.InsertOnSubmit(p);
    DataBase.SubmitChanges();

    tm.idCameras = c.idCameras;
    tm.idEcra = e.idEcra;

```

```
tm.idProcessador = p.idProcessador;  
DataBase.Telemoveis.InsertOnSubmit(tm);  
DataBase.SubmitChanges();  
}
```

```
public double SimiliridadeMarca(string marca1, string marca2)  
{  
    if (marca1 == marca2) return 1;  
    else if (marca1 == "Apple" && marca2 == "Samsung") return 0.8; //similiridade em termos  
de preço  
    else if (marca1 == "Samsung" && marca2 == "Apple") return 0.8;  
    else if (marca1 == "LG" && marca2 == "Samsung") return 0.5;  
    else if (marca1 == "Samsung" && marca2 == "LG") return 0.5;  
    else if (marca1 == "LG" && marca2 == "Apple") return 0.4;  
    else if (marca1 == "Apple" && marca2 == "LG") return 0.4;  
    else if (marca1 == "Outras" && marca2 == "LG") return 0.5;  
    else if (marca1 == "LG" && marca2 == "Outras") return 0.5;  
    else if (marca1 == "Outras" && marca2 == "Samsung") return 0.3;  
    else if (marca1 == "Samsung" && marca2 == "Outras") return 0.3;  
    else if (marca1 == "Outras" && marca2 == "Apple") return 0.1;  
    else if (marca1 == "Apple" && marca2 == "Outras") return 0.1;  
    else return 0; //nunca poderá acontecer ser 0 porque a view não permite outras opções além  
das de cima.  
}
```

```
public double SimiliridadeRam(double ram1, double ram2)  
{  
    if (ram1 >= 10 && ram2 >= 10) return 1; //acima de 10 a ram é totalmente indiferente para  
um telemovel  
    if (Math.Abs(ram1 - ram2) > 10) return 0; //nao tem semelhança nenhuma  
    else  
    {
```

```
if (ram1 <= 2 && ram2 <= 2) //objetivo: acentuar a diferença para ram's com valores  
abaixo de 2
```

```
    return (4 - Math.Abs(ram1 - ram2)) / 4; //varia de 0.5 a 1  
    //exemplo: uma diferença de 1gb para 2gb é mais substancial do que 8gb para 9gb  
    else  
        return (10 - Math.Abs(ram1 - ram2)) / 10;  
    }  
}
```

```
public double SimiliridadeMemoriaInterna(int mInterna1, int mInterna2)  
{  
    if (mInterna1 >= 256 && mInterna2 >= 256) return 1; //acima de 256 a memoria nao tem  
importancia, sao iguais em termos de preço
```

```
    if (Math.Abs(mInterna1 - mInterna2) > 256) return 0; //nao teem semelhança nenhuma  
    else  
    {  
        if (mInterna1 <= 5 && mInterna2 <= 5) //objetivo: acentuar a diferença para memorias  
baixas
```

```
        return (10 - Math.Abs(mInterna1 - mInterna2)) / 10; //varia de 0.5 a 1  
        //exemplo: uma diferença de 5gb para 15gb é mais substancial do que 50gb para 60gb  
        else  
            return (256 - Math.Abs(mInterna1 - mInterna2)) / 256;  
    }  
}
```

```
public double SimiliridademAhBateria(int mAhBateria1, int mAhBateria2)  
{  
    if (mAhBateria1 >= 5000 && mAhBateria2 >= 5000) return 1; //não existem baterias com  
mais mAh
```

```
    if (Math.Abs(mAhBateria1 - mAhBateria2) > 5000) return 0;  
    else  
    {
```

```
    if (mAhBateria1 <= 1500 && mAhBateria2 <= 1500)
        return (3000 - Math.Abs(mAhBateria1 - mAhBateria2)) / 3000; //varia de 0.5 a 1
    else
        return (5000 - Math.Abs(mAhBateria1 - mAhBateria2)) / 5000;
    }
}
```

```
public double SimilidadeResoluçãoEcra(string res1, string res2)
{
    //"640x480 (VGA)"
    //"1280x720 (HD)"
    //"1920x1080 (FULL HD)"
    //"2560x1440 (2K)"
    //"3840x2160 (4K)"
    if (res1 == res2) return 1;

    if (res1 == "640x480 (VGA)" && res2 == "1280x720 (HD)") return 0.5;
    if (res1 == "1280x720 (HD)" && res2 == "640x480 (VGA)") return 0.5;

    if (res1 == "640x480 (VGA)" && res2 == "1920x1080 (FULL HD)") return 0.4;
    if (res1 == "1920x1080 (FULL HD)" && res2 == "640x480 (VGA)") return 0.4;

    if (res1 == "640x480 (VGA)" && res2 == "2560x1440 (2K)") return 0.2;
    if (res1 == "2560x1440 (2K)" && res2 == "640x480 (VGA)") return 0.2;

    if (res1 == "640x480 (VGA)" && res2 == "3840x2160 (4K)") return 0;
    if (res1 == "3840x2160 (4K)" && res2 == "640x480 (VGA)") return 0;
    //-----

    if (res1 == "1920x1080 (FULL HD)" && res2 == "1280x720 (HD)") return 0.7;
    if (res1 == "1280x720 (HD)" && res2 == "1920x1080 (FULL HD)") return 0.7;

    if (res1 == "2560x1440 (2K)" && res2 == "1280x720 (HD)") return 0.5;
```

```
if (res1 == "1280x720 (HD)" && res2 == "2560x1440 (2K)") return 0.5;

if (res1 == "1280x720 (HD)" && res2 == "3840x2160 (4K)") return 0.2;
if (res1 == "3840x2160 (4K)" && res2 == "1280x720 (HD)") return 0.2;
//-----
if (res1 == "2560x1440 (2K)" && res2 == "1920x1080 (FULL HD)") return 0.6;
if (res1 == "1980x1080 (FULL HD)" && res2 == "2560x1440 (2K)") return 0.6;

if (res1 == "3840x2160 (4K)" && res2 == "1920x1080 (FULL HD)") return 0.4;
if (res1 == "1980x1080 (FULL HD)" && res2 == "3840x2160 (4K)") return 0.4;
//-----
if (res1 == "3840x2160 (4K)" && res2 == "2560x1440 (2K)") return 0.5;
if (res1 == "2560x1440 (2K)" && res2 == "3840x2160 (4K)") return 0.5;
else return 0;

}

public double SimiliridadeTamanhoEcrã(double tamanho1, double tamanho2)
{
    //tamanho ideal de um ecrã para telemovel segundo a maioria das sondagens: 5 polegadas
    if (tamanho1 > 10 && tamanho2 > 10) return 1;
    if (Math.Abs(tamanho1 - tamanho2) < 5) return 5 - (Math.Abs(tamanho1 - tamanho2)) / 5;
    else
    {
        if (Math.Abs(tamanho1 - tamanho2) == 5) return 1;
        else
            return (10 - Math.Abs(tamanho1 - tamanho2)) / 5;
    }
}

public double SimiliridadeVelocidadeProcessador(double vel1, double vel2)
{
    //acima de 5GHz é indiferente
```



```
if (vel1 >= 5 && vel2 >= 5) return 1;
if (Math.Abs(vel1 - vel2) > 5) return 0;
else
{
    if (vel1 <= 1500 && vel2 <= 1.5)
        return (3 - Math.Abs(vel1 - vel2)) / 3; //varia de 0.5 a 1
    else
        return (5 - Math.Abs(vel1 - vel2)) / 5000;
}
}

public double SimiliridadeNucleosProcessador(int n1, int n2)
{
    //é inutil ter mais que 16 nucleos
    if (n1 >= 16 && n2 >= 16) return 1;
    if (Math.Abs(n1 - n2) > 16) return 0;
    else
    {
        if (n1 <= 2 && n2 <= 2)
            return (4 - Math.Abs(n1 - n2)) / 4; //varia de 0.5 a 1
        else
            return (16 - Math.Abs(n1 - n2)) / 16;
    }
}

public double SimiliridadeResolucaoCameraFrontal(double cFrontal1, double cFrontal2)
{
    //no mercado nao existem cameras frontais com mais de 16 megapixels
    if (cFrontal1 >= 16 && cFrontal2 >= 16) return 1;
    if (Math.Abs(cFrontal1 - cFrontal2) > 16) return 0;
    else
    {
        if (cFrontal1 <= 4 && cFrontal2 <= 4)
            return (8 - Math.Abs(cFrontal1 - cFrontal2)) / 8; //varia de 0.5 a 1
    }
}
```

```
else
    return (16 - Math.Abs(cFrontal1 - cFrontal2)) / 16;
}
}

public double SimiliridadeResolucaoCameraTraseira(double cTraseira1, double cTraseira2)
{
    if (cTraseira1 >= 20 && cTraseira2 >= 20) return 1;
    if (Math.Abs(cTraseira1 - cTraseira2) > 20) return 0;
    else
    {
        if (cTraseira1 <= 8 && cTraseira2 <= 8)
            return (16 - Math.Abs(cTraseira1 - cTraseira2)) / 16; //varia de 0.5 a 1
        else
            return (20 - Math.Abs(cTraseira1 - cTraseira2)) / 20;
    }
}

public double SimiliridadeEstado(string estado1, string estado2)
{
    int e1 = 0, e2 = 0;
    if (estado1 == "Péssimo") e1 = 0;
    if (estado1 == "Mau") e1 = 1;
    if (estado1 == "Intermédio") e1 = 2;
    if (estado1 == "Bom") e1 = 3;
    if (estado1 == "Muito Bom") e1 = 4;
    if (estado1 == "Como Novo") e1 = 5;

    if (estado2 == "Péssimo") e2 = 0;
    if (estado2 == "Mau") e2 = 1;
    if (estado2 == "Intermédio") e2 = 2;
    if (estado2 == "Bom") e2 = 3;
```

```
if (estado2 == "Muito Bom") e2 = 4;
if (estado2 == "Como Novo") e2 = 5;

return (5 - Math.Abs(e1 - e2)) / 5;
}

public double SimiliridadeIdade(int idade1, int idade2)
{
    //dada em meses
    if (idade1 >= 48 && idade2 >= 48) return 1; //acima de 4 anos é igual
    if (Math.Abs(idade1 - idade2) > 48) return 0;
    else
    {
        if (idade1 <= 6 && idade2 <= 6) //a diferença é mais relevante
            return (12 - Math.Abs(idade1 - idade2)) / 12; //varia de 0.5 a 1
        else
            return (48 - Math.Abs(idade1 - idade2)) / 48;
    }
}

public void Avaliação()
{
    double melhorSimiliridade = 0; //ainda não foi encontrado nenhuma aproximação
    double similiridadeItem = 0;
    int idItem = 0;

    const double pesoMarca = 0.35;
    const double pesoRam = 0.075;
    const double pesoMemoriainterna = 0.05;
    const double pesomAhBateria = 0.05;
    const double pesoResEcra = 0.1;
    const double pesoTamanho = 0.025;
    const double pesoProcessadorvelocidade = 0.1;
```

```

const double pesoProcessadornucleos = 0.025;
const double pesoCamerafrontal = 0.025;
const double pesoCameratraseira = 0.05;
const double pesoEstado = 0.1;
const double pesoIdade = 0.05;

foreach (Telemovel item in DataBase.Telemovels)
{
    //pesos acima
    similaridadeItem = SimilaridadeMarca(DataBase.marcas.Single(x => x.idMarca ==
tm.idMarca).nome, item.marca.nome) * pesoMarca + SimilaridadeRam(tm.ram, item.ram) *
pesoRam + SimilaridadeMemoriaInterna(tm.memoriaInterna, item.memoriaInterna) *
pesoMemoriaInterna
        + SimilaridademAhBateria(tm.mAhBateria, item.mAhBateria) *
pesomAhBateria + SimilaridadeResoluçãoEcra(DataBase.ResoluçãoEcra.Single(x =>
x.idResoluçãoEcra == e.idResoluçãoEcra).designação, item.Ecra.ResoluçãoEcra.designação) *
pesoResEcra
        + SimilaridadeTamanhoEcra(e.tamanho, item.Ecra.tamanho) *
pesoTamanho + SimilaridadeVelocidadeProcessador(p.velocidadeProcessador,
item.Processador.velocidadeProcessador) * pesoProcessadorvelocidade
        + SimilaridadeNucleosProcessador(p.nucleosProcessador,
item.Processador.nucleosProcessador) * pesoProcessadornucleos +
SimilaridadeResolucaoCameraFrontal(c.resolucaoFrontal, item.Camera.resolucaoFrontal) *
pesoCamerafrontal
        + SimilaridadeResolucaoCameraTraseira(c.resolucaoTraseira,
item.Camera.resolucaoTraseira) * pesoCameratraseira +
SimilaridadeEstado(DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação,
item.Estado.designação) * pesoEstado
        + SimilaridadeIdade(tm.idade, item.idade) * pesoIdade;

    if (similaridadeItem > melhorSimilaridade)
    {

```

```

        melhorSimiliridade = similaridadeItem;
        tm.valorFinal = Convert.ToInt32(item.valorFinal); //int da base de dados nao é
compativel com int do visual studio
        idItem = item.idTelemovel;
    }
}
submeterNaDB();

if (similaridadeItem != 1)
    acertarPreço(DataBase.Telemovels.Single(x => x.idTelemovel == idItem));

tm = new Telemovel(); //reinicializar para a proxima avaliação
e = new Ecra();
c = new Camera();
p = new Processador();
}
public void acertarPreço(Telemovel caso)
{
    List<Telemovel> listT = DataBase.Telemovels.OrderByDescending(p =>
p.idTelemovel).ToList();
    int idT = listT.First().idTelemovel; //para buscarmos o ultimo a ser inserido (primeiro nesta
lista porque está ordenada em descending)
    if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome != caso.marca.nome)
    {
        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" &&
caso.marca.nome == "Samsung") DataBase.Telemovels.Single(x => x.idTelemovel ==
idT).valorFinal += (int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal *
0.1);

        if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "Apple") DataBase.Telemovels.Single(x => x.idTelemovel ==
idT).valorFinal -= (int)(DataBase.Telemovels.Single(x => x.idTelemovel == idT).valorFinal *
0.1);

```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" &&
caso.marca.nome == "Samsung") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal -= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.2);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "LG") DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal
+= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.2);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" &&
caso.marca.nome == "Apple") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal -= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.3);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" &&
caso.marca.nome == "LG") DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal
+= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.3);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" &&
caso.marca.nome == "LG") DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -
= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.3);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "LG" &&
caso.marca.nome == "Outras") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal += (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.3);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" &&
caso.marca.nome == "Samsung") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal -= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.4);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Samsung" &&
caso.marca.nome == "Outras") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal += (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.4);
```

```
if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Outras" &&
caso.marca.nome == "Apple") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
```

```
idT).valorFinal -= (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.6);

    if (DataBase.marcas.Single(x => x.idMarca == tm.idMarca).nome == "Apple" &&
caso.marca.nome == "Outras") DataBase.Telemoveis.Single(x => x.idTelemoveis ==
idT).valorFinal += (int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal *
0.6);

    }

    if (tm.ram > caso.ram)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimiliridadeRam(tm.ram, caso.ram)) * 0.05);
    }

    if (tm.ram < caso.ram)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimiliridadeRam(tm.ram, caso.ram)) * 0.05);
    }

    if (tm.memoriaInterna > caso.memoriaInterna)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimiliridadeMemoriaInterna(tm.memoriaInterna, caso.memoriaInterna)) * 0.025);
    }

    if (tm.memoriaInterna < caso.memoriaInterna)
    {
```

```

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeMemoriaInterna(tm.memoriaInterna, caso.memoriaInterna)) * 0.025);
    }

    if (tm.mAhBateria > caso.mAhBateria)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidademAhBateria(tm.mAhBateria, caso.mAhBateria)) * 0.05);
    }

    if (tm.mAhBateria < caso.mAhBateria)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidademAhBateria(tm.mAhBateria, caso.mAhBateria)) * 0.05);
    }

    if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcra ==
e.idResolucaoEcra).designacao != caso.Ecra.ResolucaoEcra.designacao)
    {
        if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcra ==
e.idResolucaoEcra).designacao == "640x480 (VGA)" && caso.Ecra.ResolucaoEcra.designacao
== "1280x720 (HD)")
        {
            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.05);
            if (DataBase.ResolucaoEcras.Single(x => x.idResolucaoEcra ==
e.idResolucaoEcra).designacao == "1280x720 (HD)" && caso.Ecra.ResolucaoEcra.designacao ==
"640x480 (VGA)")
            {
                DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.05);
            }
        }
    }

```



```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação
== "1920x1080 (FULL HD)")

```

```

    DataBase.Telemoveis.Single(x  =>  x.idTelemovel  ==  idT).valorFinal  -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.08);

```

```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação      ==      "1920x1080      (FULL      HD)"      &&
caso.Ecra.ResoluçãoEcra.designação == "640x480 (VGA)")

```

```

    DataBase.Telemoveis.Single(x  =>  x.idTelemovel  ==  idT).valorFinal  +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.08);

```

```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação
== "2560x1440 (2K)")

```

```

    DataBase.Telemoveis.Single(x  =>  x.idTelemovel  ==  idT).valorFinal  -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.1);

```

```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação
== "640x480 (VGA)")

```

```

    DataBase.Telemoveis.Single(x  =>  x.idTelemovel  ==  idT).valorFinal  +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.1);

```

```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "640x480 (VGA)" && caso.Ecra.ResoluçãoEcra.designação
== "3840x2160 (4K)")

```

```

    DataBase.Telemoveis.Single(x  =>  x.idTelemovel  ==  idT).valorFinal  -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.15);

```

```

if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação
== "640x480 (VGA)")

```

```

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.15);

//-----

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "1920x1080 (FULL HD)" &&
caso.Ecra.ResoluçãoEcra.designação == "1280x720 (HD)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.06);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "1280x720 (HD)" && caso.Ecra.ResoluçãoEcra.designação ==
"1920x1080 (FULL HD)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.06);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação
== "1280x720 (HD)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "1280x720 (HD)" && caso.Ecra.ResoluçãoEcra.designação ==
"2560x1440 (2K)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.08);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "1280x720 (HD)" && caso.Ecra.ResoluçãoEcra.designação ==
"3840x2160 (4K)")

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.12);

```

```

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação
== "1280x720 (HD)")

            DataBase.Telemoveis.Single(x      =>      x.idTelemovel      ==      idT).valorFinal      +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.12);

//-----

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação
== "1920x1080 (FULL HD)")

            DataBase.Telemoveis.Single(x      =>      x.idTelemovel      ==      idT).valorFinal      +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.06);

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação      ==      "1980x1080      (FULL      HD)"      &&
caso.Ecra.ResoluçãoEcra.designação == "2560x1440 (2K)")

            DataBase.Telemoveis.Single(x      =>      x.idTelemovel      ==      idT).valorFinal      -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.06);

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação
== "1920x1080 (FULL HD)")

            DataBase.Telemoveis.Single(x      =>      x.idTelemovel      ==      idT).valorFinal      +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.08);

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação      ==      "1980x1080      (FULL      HD)"      &&
caso.Ecra.ResoluçãoEcra.designação == "3840x2160 (4K)")

            DataBase.Telemoveis.Single(x      =>      x.idTelemovel      ==      idT).valorFinal      -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemovel == idT).valorFinal * 0.08);

//-----

        if      (DataBase.ResoluçãoEcras.Single(x      =>      x.idResoluçãoEcra      ==
e.idResoluçãoEcra).designação == "3840x2160 (4K)" && caso.Ecra.ResoluçãoEcra.designação
== "2560x1440 (2K)")

```

```

        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.04);

        if (DataBase.ResoluçãoEcras.Single(x => x.idResoluçãoEcra ==
e.idResoluçãoEcra).designação == "2560x1440 (2K)" && caso.Ecra.ResoluçãoEcra.designação
== "3840x2160 (4K)")

            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
(int)(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * 0.04);

    }

    if (e.tamanho > caso.Ecra.tamanho)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeTamanhoEcra(e.tamanho, caso.Ecra.tamanho)) * 0.025);
    }

    if (e.tamanho < caso.Ecra.tamanho)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeTamanhoEcra(e.tamanho, caso.Ecra.tamanho)) * 0.025);
    }

    if (p.velocidadeProcessador > caso.Processador.velocidadeProcessador)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeVelocidadeProcessador(p.velocidadeProcessador,
caso.Processador.velocidadeProcessador)) * 0.05);
    }

```

```

if (p.velocidadeProcessador < caso.Processador.velocidadeProcessador)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeVelocidadeProcessador(p.velocidadeProcessador,
caso.Processador.velocidadeProcessador)) * 0.05);
}

if (p.nucleosProcessador > caso.Processador.nucleosProcessador)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeNucleosProcessador(p.nucleosProcessador, caso.Processador.nucleosProcessador)) *
0.025);
}

if (p.nucleosProcessador < caso.Processador.nucleosProcessador)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeNucleosProcessador(p.nucleosProcessador, caso.Processador.nucleosProcessador)) *
0.025);
}

if (c.resolucaoFrontal > caso.Camera.resolucaoFrontal)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeResolucaoCameraFrontal(c.resolucaoFrontal, caso.Camera.resolucaoFrontal)) *
0.05);
}

```

```

if (c.resolucaoFrontal < caso.Camera.resolucaoFrontal)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeResolucaoCameraFrontal(c.resolucaoFrontal, caso.Camera.resolucaoFrontal)) *
0.05);
}

if (c.resolucaoTraseira > caso.Camera.resolucaoTraseira)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeResolucaoCameraTraseira(c.resolucaoTraseira, caso.Camera.resolucaoTraseira)) *
0.05);
}

if (c.resolucaoTraseira < caso.Camera.resolucaoTraseira)
{
    DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeResolucaoCameraTraseira(c.resolucaoTraseira, caso.Camera.resolucaoTraseira)) *
0.05);
}

if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação !=
caso.Estado.designação)
{
    int e1 = 0, e2 = 0;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Péssimo")
e1 = 0;
    if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Mau") e1
= 1;

```

```

        if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação ==
"Intermédio") e1 = 2;

        if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Bom") e1
= 3;

        if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Muito
Bom") e1 = 4;

        if (DataBase.Estados.Single(x => x.idEstado == tm.idEstado).designação == "Como
Novo") e1 = 5;


        if (caso.Estado.designação == "Péssimo") e2 = 0;
        if (caso.Estado.designação == "Mau") e2 = 1;
        if (caso.Estado.designação == "Intermédio") e2 = 2;
        if (caso.Estado.designação == "Bom") e2 = 3;
        if (caso.Estado.designação == "Muito Bom") e2 = 4;
        if (caso.Estado.designação == "Como Novo") e2 = 5;


        int resultado = (5 - Math.Abs(e1 - e2)) / 5;
        if (e1 > e2) //impossível ser igual, verificado antes

            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * ((1 -
resultado) * 0.1));
        else

            DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * ((1 -
resultado) * 0.1));
    }

    if (tm.idade > caso.idade)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal +=
Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
SimilidadeIdade(tm.idade, caso.idade)) * 0.1);

```

```

    }

    if (tm.idade < caso.idade)
    {
        DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal -=
        Convert.ToInt32(DataBase.Telemoveis.Single(x => x.idTelemoveis == idT).valorFinal * (1 -
        SimiliridadeIdade(tm.idade, caso.idade)) * 0.1);
    }
    DataBase.SubmitChanges();
}
}
}

```

HomeController.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using Modulo2IA_CBR_Telemoveis.Models;

namespace Modulo2IA_CBR_Telemoveis.Controllers
{
    public class HomeController : Controller
    {
        Data D = new Data();
        // GET: Home
        public ActionResult Avaliaç o()
        {
            ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
            ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado", "designaç o");

```



```
ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "idResoluçãoEcra",  
"designação");  
return View();  
}
```

[HttpPost]

```
public ActionResult Avaliação(FormCollection collection)  
{
```

```
    if (string.IsNullOrEmpty(collection["idMarca"]))
```

```
        ModelState.AddModelError("idMarca", "Falta preencher este ou mais campos!");
```

```
    if (string.IsNullOrEmpty(collection["ram"]))
```

```
        ModelState.AddModelError("ram", "Falta preencher este ou mais campos!");
```

```
    else
```

```
    {
```

```
        try
```

```
        {
```

```
            if
```

```
                (Convert.ToDouble(collection["ram"],
```

```
System.Globalization.CultureInfo.InvariantCulture) <= 0)
```

```
                ModelState.AddModelError("ram", "Este campo tem de ter valores positivos!");
```

```
        }
```

```
    catch
```

```
    {
```

```
        ModelState.AddModelError("ram", "Valor inválido.");
```

```
    }
```

```
}
```

```
    if (string.IsNullOrEmpty(collection["interna"]))
```

```
        ModelState.AddModelError("interna", "Falta preencher este ou mais campos!");
```

```
    else
```

```
    {
```

```
try
{
    if (Convert.ToInt32(collection["interna"]) <= 0)
        ModelState.AddModelError("interna", "Este campo tem de ter valores positivos!");
}
catch
{
    ModelState.AddModelError("interna", "Valor inválido.");
}
}

if (string.IsNullOrEmpty(collection["Ecra.idResoluçãoEcra"]))
    ModelState.AddModelError("Ecra.idResoluçãoEcra", "Falta preencher este ou mais campos!");

if (string.IsNullOrEmpty(collection["tamanho"]))
    ModelState.AddModelError("tamanho", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["tamanho"],
            System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("tamanho", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("tamanho", "Valor inválido.");
    }
}
```

```
if (string.IsNullOrEmpty(collection["velocidade"]))
    ModelState.AddModelError("velocidade", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["velocidade"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("velocidade", "Este campo tem de ter valores
positivos!");
    }
    catch
    {
        ModelState.AddModelError("velocidade", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["nucleos"]))
    ModelState.AddModelError("nucleos", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["nucleos"]) <= 0)
            ModelState.AddModelError("nucleos", "Este campo tem de ter valores
positivos!");
    }
    catch
    {
        ModelState.AddModelError("nucleos", "Valor inválido.");
    }
}
```

```
if (string.IsNullOrEmpty(collection["frontal"]))
    ModelState.AddModelError("frontal", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["frontal"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("frontal", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("frontal", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["traseira"]))
    ModelState.AddModelError("traseira", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["traseira"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("traseira", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("traseira", "Valor inválido.");
    }
}
```

```
if (string.IsNullOrEmpty(collection["bateria"]))
    ModelState.AddModelError("bateria", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["bateria"]) <= 0)
            ModelState.AddModelError("bateria", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("bateria", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["idEstado"]))
    ModelState.AddModelError("idEstado", "Falta preencher este ou mais campos!");

if (string.IsNullOrEmpty(collection["idade"]))
    ModelState.AddModelError("idade", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["idade"]) <= 0)
            ModelState.AddModelError("idade", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("idade", "Valor inválido.");
    }
}
```

```

    }

    ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
    ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado", "designação");
    ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "idResoluçãoEcra",
    "designação");

    if (ModelState.IsValid)
    {
        D.SetTelemovel(Convert.ToInt32(collection["idMarca"]),
        Convert.ToInt32(collection["Ecra.idResoluçãoEcra"]), Convert.ToInt32(collection["idEstado"]),
        Convert.ToDouble(collection["ram"],
        System.Globalization.CultureInfo.InvariantCulture), Convert.ToInt32(collection["interna"]),
        Convert.ToInt32(collection["bateria"]), Convert.ToDouble(collection["tamanho"],
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToDouble(collection["velocidade"],
        System.Globalization.CultureInfo.InvariantCulture), Convert.ToInt32(collection["nucleos"]),
        Convert.ToDouble(collection["frontal"],
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToDouble(collection["traseira"], System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToInt32(collection["idade"]));

        D.Avaliação();

        int i = D.DataBase.Telemoveis.OrderByDescending(p =>
        p.idTelemovel).ToList().First().idTelemovel;

        ViewBag.Avaliação = "Avaliação: " + D.DataBase.Telemoveis.Single(x =>
        x.idTelemovel == i).valorFinal + " €";

        return View("ValorAvaliação", D.DataBase.Telemoveis.Single(x => x.idTelemovel ==
        i));
    }
    else

```

```
{
    return View();
}

public ActionResult ValorAvaliação()
{
    return View();
}

public ActionResult AdicionarCaso()
{
    ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
    ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado", "designação");
    ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "idResoluçãoEcra",
"designação");
    return View();
}

[HttpPost]
public ActionResult AdicionarCaso(FormCollection collection)
{
    if (string.IsNullOrEmpty(collection["idMarca"]))
        ModelState.AddModelError("idMarca", "Falta preencher este ou mais campos!");

    if (string.IsNullOrEmpty(collection["ram"]))
        ModelState.AddModelError("ram", "Falta preencher este ou mais campos!");
    else
    {
        try
        {
            if (Convert.ToDouble(collection["ram"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
                ModelState.AddModelError("ram", "Este campo tem de ter valores positivos!");
        }
    }
}
```

```
}  
catch  
{  
    ModelState.AddModelError("ram", "Valor inválido.");  
}  
}  
  
if (string.IsNullOrEmpty(collection["interna"]))  
    ModelState.AddModelError("interna", "Falta preencher este ou mais campos!");  
else  
{  
    try  
    {  
        if (Convert.ToInt32(collection["interna"]) <= 0)  
            ModelState.AddModelError("interna", "Este campo tem de ter valores positivos!");  
    }  
    catch  
    {  
        ModelState.AddModelError("interna", "Valor inválido.");  
    }  
}  
  
if (string.IsNullOrEmpty(collection["Ecra.idResoluçãoEcra"]))  
    ModelState.AddModelError("Ecra.idResoluçãoEcra", "Falta preencher este ou mais  
campos!");  
  
if (string.IsNullOrEmpty(collection["tamanho"]))  
    ModelState.AddModelError("tamanho", "Falta preencher este ou mais campos!");  
else  
{  
    try  
    {
```



```
        if (Convert.ToDouble(collection["tamanho"],  
System.Globalization.CultureInfo.InvariantCulture) <= 0)  
            ModelState.AddModelError("tamanho", "Este campo tem de ter valores  
positivos!");  
    }  
    catch  
    {  
        ModelState.AddModelError("tamanho", "Valor inválido.");  
    }  
}
```

```
if (string.IsNullOrEmpty(collection["velocidade"]))  
    ModelState.AddModelError("velocidade", "Falta preencher este ou mais campos!");  
else  
{  
    try  
    {  
        if (Convert.ToDouble(collection["velocidade"],  
System.Globalization.CultureInfo.InvariantCulture) <= 0)  
            ModelState.AddModelError("velocidade", "Este campo tem de ter valores  
positivos!");  
    }  
    catch  
    {  
        ModelState.AddModelError("velocidade", "Valor inválido.");  
    }  
}
```

```
if (string.IsNullOrEmpty(collection["nucleos"]))  
    ModelState.AddModelError("nucleos", "Falta preencher este ou mais campos!");  
else  
{
```

```
try
{
    if (Convert.ToInt32(collection["nucleos"]) <= 0)
        ModelState.AddModelError("nucleos", "Este campo tem de ter valores positivos!");
}
catch
{
    ModelState.AddModelError("nucleos", "Valor inválido.");
}
}

if (string.IsNullOrEmpty(collection["frontal"]))
    ModelState.AddModelError("frontal", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["frontal"],
            System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("frontal", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("frontal", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["traseira"]))
    ModelState.AddModelError("traseira", "Falta preencher este ou mais campos!");
else
{
```

```
try
{
    if (Convert.ToDouble(collection["traseira"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
        ModelState.AddModelError("traseira", "Este campo tem de ter valores positivos!");
}
catch
{
    ModelState.AddModelError("traseira", "Valor inválido.");
}
}

if (string.IsNullOrEmpty(collection["bateria"]))
    ModelState.AddModelError("bateria", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["bateria"]) <= 0)
            ModelState.AddModelError("bateria", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("bateria", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["idEstado"]))
    ModelState.AddModelError("idEstado", "Falta preencher este ou mais campos!");

if (string.IsNullOrEmpty(collection["idade"]))
    ModelState.AddModelError("idade", "Falta preencher este ou mais campos!");
```

```
else
{
    try
    {
        if (Convert.ToInt32(collection["idade"]) <= 0)
            ModelState.AddModelError("idade", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("idade", "Valor inválido.");
    }
}
```

```
if (string.IsNullOrEmpty(collection["valor"]))
    ModelState.AddModelError("valor", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["valor"]) <= 0)
            ModelState.AddModelError("valor", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("valor", "Valor inválido.");
    }
}
```

```
ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado", "designação");
ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "IdResoluçãoEcra",
"designação");
```

```
if (ModelState.IsValid)
{
    D.SetTelemovel(Convert.ToInt32(collection["idMarca"]),
    Convert.ToInt32(collection["Ecra.idResoluçãoEcra"]), Convert.ToInt32(collection["idEstado"]),
    Convert.ToDouble(collection["ram"]),
    System.Globalization.CultureInfo.InvariantCulture), Convert.ToInt32(collection["interna"]),
    Convert.ToInt32(collection["bateria"]),    Convert.ToDouble(collection["tamanho"],
    System.Globalization.CultureInfo.InvariantCulture),
    Convert.ToDouble(collection["velocidade"],
    System.Globalization.CultureInfo.InvariantCulture),    Convert.ToInt32(collection["nucleos"]),
    Convert.ToDouble(collection["frontal"],
    System.Globalization.CultureInfo.InvariantCulture),
    Convert.ToDouble(collection["traseira"],    System.Globalization.CultureInfo.InvariantCulture),
    Convert.ToInt32(collection["idade"]));

    D.tm.valorFinal = Convert.ToInt32(collection["valor"]);
    D.AdicionarCaso();

    TempData["Caso_Adicionado"] = "Caso adicionado com sucesso!";
    return View();
}
else
{
    return View();
}
}

// GET: Home/Details/5
public ActionResult ListarCasos()
{
    return View(D.DataBase.Telemovels);
}
```

```
// GET: Home/Edit/5
public ActionResult EditarCaso(int id)
{
    ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
    ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado", "designação");
    ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "idResoluçãoEcra",
"designação");
    return View(D.DataBase.Telemoveis.Single(x => x.idTelemovel == id));
}

// POST: Home/Edit/5
[HttpPost]
public ActionResult EditarCaso(int id, FormCollection collection)
{
    if (string.IsNullOrEmpty(collection["idMarca"]))
        ModelState.AddModelError("idMarca", "Falta preencher este ou mais campos!");

    if (string.IsNullOrEmpty(collection["ram"]))
        ModelState.AddModelError("ram", "Falta preencher este ou mais campos!");
    else
    {
        try
        {
            if (Convert.ToDouble(collection["ram"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
                ModelState.AddModelError("ram", "Este campo tem de ter valores positivos!");
        }
        catch
        {

```

```
ModelState.AddModelError("ram", "Valor inválido.");
}
}

if (string.IsNullOrEmpty(collection["memoriaInterna"]))
    ModelState.AddModelError("memoriaInterna", "Falta preencher este ou mais
campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["memoriaInterna"]) <= 0)
            ModelState.AddModelError("memoriaInterna", "Este campo tem de ter valores
positivos!");
    }
    catch
    {
        ModelState.AddModelError("memoriaInterna", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["Ecra.idResoluçãoEcra"]))
    ModelState.AddModelError("Ecra.idResoluçãoEcra", "Falta preencher este ou mais
campos!");

if (string.IsNullOrEmpty(collection["Ecra.tamanho"]))
    ModelState.AddModelError("Ecra.tamanho", "Falta preencher este ou mais campos!");
else
{
    try
    {
```

```
        if (Convert.ToDouble(collection["Ecra.tamanho"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("Ecra.tamanho", "Este campo tem de ter valores
positivos!");
        }
        catch
        {
            ModelState.AddModelError("Ecra.tamanho", "Valor inválido.");
        }
    }

    if (string.IsNullOrEmpty(collection["Processador.velocidadeProcessador"]))
        ModelState.AddModelError("Processador.velocidadeProcessador", "Falta preencher
este ou mais campos!");
    else
    {
        try
        {
            if (Convert.ToDouble(collection["Processador.velocidadeProcessador"],
System.Globalization.CultureInfo.InvariantCulture) <= 0)
                ModelState.AddModelError("Processador.velocidadeProcessador", "Este campo
tem de ter valores positivos!");
        }
        catch
        {
            ModelState.AddModelError("Processador.velocidadeProcessador", "Valor
inválido.");
        }
    }

    if (string.IsNullOrEmpty(collection["Processador.nucleosProcessador"]))
```



```
ModelState.AddModelError("Processador.nucleosProcessador", "Falta preencher este  
ou mais campos!");  
else  
{  
    try  
    {  
        if (Convert.ToInt32(collection["Processador.nucleosProcessador"]) <= 0)  
            ModelState.AddModelError("Processador.nucleosProcessador", "Este campo tem  
de ter valores positivos!");  
    }  
    catch  
    {  
        ModelState.AddModelError("Processador.nucleosProcessador", "Valor inválido.");  
    }  
}  
  
if (string.IsNullOrEmpty(collection["Camera.resolucaoFrontal"]))  
    ModelState.AddModelError("Camera.resolucaoFrontal", "Falta preencher este ou mais  
campos!");  
else  
{  
    try  
    {  
        if (Convert.ToDouble(collection["Camera.resolucaoFrontal"],  
System.Globalization.CultureInfo.InvariantCulture) <= 0)  
            ModelState.AddModelError("Camera.resolucaoFrontal", "Este campo tem de ter  
valores positivos!");  
    }  
    catch  
    {  
        ModelState.AddModelError("Camera.resolucaoFrontal", "Valor inválido.");  
    }  
}
```

```
}

if (string.IsNullOrEmpty(collection["Camera.resolucaoTraseira"]))
    ModelState.AddModelError("Camera.resolucaoTraseira", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["Camera.resolucaoTraseira"],
            System.Globalization.CultureInfo.InvariantCulture) <= 0)
            ModelState.AddModelError("Camera.resolucaoTraseira", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("Camera.resolucaoTraseira", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["mAhBateria"]))
    ModelState.AddModelError("bateria", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToDouble(collection["mAhBateria"]) <= 0)
            ModelState.AddModelError("mAhBateria", "Este campo tem de ter valores positivos!");
    }
    catch
    {

```

```
ModelState.AddModelError("mAhBateria", "Valor inválido.");
}
}

if (string.IsNullOrEmpty(collection["idEstado"]))
    ModelState.AddModelError("idEstado", "Falta preencher este ou mais campos!");

if (string.IsNullOrEmpty(collection["idade"]))
    ModelState.AddModelError("idade", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["idade"]) <= 0)
            ModelState.AddModelError("idade", "Este campo tem de ter valores positivos!");
    }
    catch
    {
        ModelState.AddModelError("idade", "Valor inválido.");
    }
}

if (string.IsNullOrEmpty(collection["valorFinal"]))
    ModelState.AddModelError("valorFinal", "Falta preencher este ou mais campos!");
else
{
    try
    {
        if (Convert.ToInt32(collection["valorFinal"]) <= 0)
            ModelState.AddModelError("valorFinal", "Este campo tem de ter valores positivos!");
    }
}
```

```
        catch
        {
            ModelState.AddModelError("valorFinal", "Valor inválido.");
        }
    }
    if (ModelState.IsValid)
    {
        D.EditarCaso(id, Convert.ToInt32(collection["idMarca"]),
        Convert.ToInt32(collection["Ecra.idResoluçãoEcra"]), Convert.ToInt32(collection["idEstado"]),
        Convert.ToDouble(collection["ram"]),
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToInt32(collection["memoriaInterna"]),
        Convert.ToInt32(collection["mAhBateria"]),
        Convert.ToDouble(collection["Ecra.tamanho"]),
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToDouble(collection["Processador.velocidadeProcessador"],
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToInt32(collection["Processador.nucleosProcessador"]),
        Convert.ToDouble(collection["Camera.resolucaoFrontal"],
        System.Globalization.CultureInfo.InvariantCulture),
        Convert.ToDouble(collection["Camera.resolucaoTraseira"],
        System.Globalization.CultureInfo.InvariantCulture), Convert.ToInt32(collection["idade"]),
        Convert.ToInt32(collection["valorFinal"]));
        TempData["Caso_Editado"] = "Caso Editado com sucesso!";
        return View("ListarCasos", D.DataBase.Telemoveis);
    }
    else
    {
        ViewBag.ListaMarcas = new SelectList(D.DataBase.marcas, "idMarca", "nome");
        ViewBag.ListaEstados = new SelectList(D.DataBase.Estados, "idEstado",
        "designação");
    }
}
```

```

        ViewBag.ListaRes = new SelectList(D.DataBase.ResoluçãoEcras, "idResoluçãoEcra",
"designação");
        return View();
    }
}

// GET: Home/Delete/5
public ActionResult EliminarCaso(int id)
{
    return View(D.DataBase.Telemoveis.Single(x => x.idTelemovel == id));
}

// POST: Home/Delete/5
[HttpPost]
public ActionResult EliminarCaso(int id, FormCollection C)
{
    try
    {
        int idecra = D.DataBase.Telemoveis.Single(x => x.idTelemovel == id).idEcra;
        int idpro = D.DataBase.Telemoveis.Single(x => x.idTelemovel == id).idProcessador;
        int idcam = D.DataBase.Telemoveis.Single(x => x.idTelemovel == id).idCameras;
        D.DataBase.Telemoveis.DeleteOnSubmit(D.DataBase.Telemoveis.Single(x
x.idTelemovel == id));
        D.DataBase.SubmitChanges();

        D.DataBase.Ecras.DeleteOnSubmit(D.DataBase.Ecras.Single(x => x.idEcra ==
idecra));
        D.DataBase.Cameras.DeleteOnSubmit(D.DataBase.Cameras.Single(x => x.idCameras
== idcam));
        D.DataBase.Processadors.DeleteOnSubmit(D.DataBase.Processadors.Single(x
x.idProcessador == idpro));
        D.DataBase.SubmitChanges();
    }
}

```

```
        TempData["Caso_Eliminado"] = "Caso eliminado com sucesso!";
    }
    catch
    {
        TempData["Caso_Não_Eliminado"] = "Erro: Caso não eliminado!";
    }
    return View("ListarCasos", D.DataBase.Telemoveis);

}

public ActionResult Contactos()
{
    return View();
}

public ActionResult Sobre()
{
    return View();
}
}
}
```

7.2. Código SQL – Definição da Base de Dados

```
USE MASTER
```

```
GO
```

```
CREATE DATABASE CBRTelemoveis
```

```
GO
```

```
USE CBRTelemoveis
```

```
GO
```

```
CREATE TABLE ResoluçãoEcra(  
idResoluçãoEcra INTEGER NOT NULL IDENTITY(1,1),  
designação VARCHAR(20) NOT NULL,  
PRIMARY KEY(idResoluçãoEcra)  
)
```

```
CREATE TABLE Ecra(  
idEcra INTEGER NOT NULL IDENTITY(1,1),  
tamanho FLOAT NOT NULL, --polegadas  
idResoluçãoEcra INTEGER NOT NULL,  
CHECK(tamanho>0),  
PRIMARY KEY(idEcra),  
FOREIGN KEY(idResoluçãoEcra) REFERENCES ResoluçãoEcra  
)
```

```
CREATE TABLE Processador(  
idProcessador INTEGER NOT NULL IDENTITY(1,1),  
velocidadeProcessador FLOAT NOT NULL, --GHz  
nucleosProcessador INTEGER NOT NULL,  
CHECK(velocidadeProcessador>0),  
CHECK(nucleosProcessador>0),  
PRIMARY KEY(idProcessador)  
)
```

```
CREATE TABLE Cameras(  
idCameras INTEGER NOT NULL IDENTITY(1,1),  
resolucaoFrontal FLOAT NOT NULL,  
resolucaoTraseira FLOAT NOT NULL,  
CHECK(resolucaoFrontal>0),  
CHECK(resolucaoTraseira>0),  
PRIMARY KEY(idCameras)  
)
```

```
CREATE TABLE Estado(  
idEstado INTEGER NOT NULL IDENTITY(1,1),  
designação VARCHAR(30) NOT NULL,  
PRIMARY KEY(idEstado)  
)
```

```
CREATE TABLE marca(  
idMarca INTEGER NOT NULL IDENTITY(1,1),  
nome VARCHAR(20) NOT NULL,  
PRIMARY KEY(idMarca)  
)
```

```
CREATE TABLE Telemovel(  
idTelemovel INTEGER NOT NULL IDENTITY(1,1),  
ram FLOAT NOT NULL,  
memoriaInterna INTEGER NOT NULL,  
mAhBateria INTEGER NOT NULL, --ex: 2000 mAh  
idade INTEGER NOT NULL,  
valorFinal INTEGER,  
idEcra INTEGER NOT NULL,  
idProcessador INTEGER NOT NULL,  
idCameras INTEGER NOT NULL,  
idEstado INTEGER NOT NULL,  
idMarca INTEGER NOT NULL,  
CHECK(ram>0),  
CHECK(memoriaInterna>0),  
CHECK(mAhBateria>0),  
PRIMARY KEY(idTelemovel),  
FOREIGN KEY(idEcra) REFERENCES Ecra,  
FOREIGN KEY(idProcessador) REFERENCES Processador,  
FOREIGN KEY(idCameras) REFERENCES Cameras,  
FOREIGN KEY(idEstado) REFERENCES Estado,
```


FOREIGN KEY(idMarca) REFERENCES Marca

)

7.3. Código SQL – Dados

SET IDENTITY_INSERT [dbo].[Telemovel] ON

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (19, 4, 64, 3000, 2, 450, 15, 15, 15, 6, 1)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (20, 1.3, 8, 2100, 26, 120, 16, 16, 16, 4, 4)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (21, 2, 16, 2100, 6, 200, 17, 17, 17, 5, 2)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (22, 2, 8, 2500, 6, 140, 18, 18, 18, 5, 3)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (23, 2, 8, 2100, 7, 100, 19, 19, 19, 5, 3)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (24, 2, 8, 2000, 6, 200, 20, 20, 20, 5, 2)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (26, 2, 6, 2300, 12, 90, 22, 22, 22, 3, 4)

INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria], [idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (27, 2, 8, 2150, 2, 150, 23, 23, 23, 5, 3)

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES (36,  
2, 16, 2200, 6, 265, 32, 32, 32, 3, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1036, 3, 16, 2500, 13, 211, 1032, 1032, 1032, 5, 1)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1038, 2, 8, 2000, 6, 200, 1034, 1034, 1034, 5, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1039, 2, 32, 2300, 2, 207, 1035, 1035, 1035, 5, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1040, 2, 8, 2300, 2, 155, 1036, 1036, 1036, 5, 3)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1041, 2, 16, 2100, 14, 224, 1037, 1037, 1037, 5, 1)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1044, 2, 8, 2500, 3, 183, 1040, 1040, 1040, 4, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1045, 1, 8, 2000, 4, 96, 1041, 1041, 1041, 4, 4)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1047, 1, 4, 2000, 22, 76, 1043, 1043, 1043, 2, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1048, 10, 200, 1900, 1, 142, 1044, 1044, 1044, 5, 4)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1049, 2, 8, 2000, 4, 138, 1045, 1045, 1045, 3, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1050, 2, 8, 1900, 2, 84, 1046, 1046, 1046, 5, 4)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1051, 1, 4, 1900, 17, 70, 1047, 1047, 1047, 3, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1053, 1, 5, 2100, 6, 95, 1049, 1049, 1049, 4, 3)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1054, 2, 4, 2100, 7, 213, 1050, 1050, 1050, 5, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1055, 3, 25, 3000, 4, 83, 1051, 1051, 1051, 3, 4)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1058, 2, 16, 2200, 4, 297, 1054, 1054, 1054, 5, 2)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1061, 2, 32, 250, 3, 393, 1057, 1057, 1057, 4, 1)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1064, 3, 64, 3200, 2, 481, 1060, 1060, 1060, 4, 1)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1065, 2, 16, 2100, 6, 206, 1061, 1061, 1061, 5, 1)
```

```
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1066, 2, 16, 2100, 6, 128, 1062, 1062, 1062, 5, 4)  
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1067, 2, 16, 2500, 24, 222, 1063, 1063, 1063, 5, 1)  
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1068, 3, 25, 3000, 4, 222, 1064, 1064, 1064, 5, 1)  
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1069, 3, 25, 3000, 4, 96, 1065, 1065, 1065, 5, 4)  
INSERT INTO [dbo].[Telemovel] ([idTelemovel], [ram], [memoriaInterna], [mAhBateria],  
[idade], [valorFinal], [idEcra], [idProcessador], [idCameras], [idEstado], [idMarca]) VALUES  
(1070, 2, 8, 2100, 10, 98, 1066, 1066, 1066, 4, 3)  
SET IDENTITY_INSERT [dbo].[Telemovel] OFF
```

```
SET IDENTITY_INSERT [dbo].[Cameras] ON  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(8, 2, 2)  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(10, 222, 22)  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(11, 222, 22)  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(12, 22, 22)  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(14, 99, 99)  
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(15, 12, 16)
```

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(16, 1.3, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(17, 4, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(18, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(19, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(20, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(21, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(22, 4, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(23, 5, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(32, 4, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1032, 2, 16)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1034, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1035, 5, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1036, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1037, 5, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1040, 8, 16)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1041, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1043, 1.3, 5)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1044, 2, 10)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1045, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1046, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1047, 2, 5)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1049, 2, 5)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1050, 2, 5)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1051, 8, 16)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1054, 2, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1057, 8, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1060, 8, 16)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1061, 8, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1062, 8, 12)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1063, 5, 8)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1064, 8, 16)

INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES
(1065, 8, 16)

```
INSERT INTO [dbo].[Cameras] ([idCameras], [resolucaoFrontal], [resolucaoTraseira]) VALUES  
(1066, 5, 8)
```

```
SET IDENTITY_INSERT [dbo].[Cameras] OFF
```

```
SET IDENTITY_INSERT [dbo].[Ecra] ON
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (15, 6, 5)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (16, 4.7, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (17, 5.5, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (18, 5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (19, 5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (20, 2, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (22, 5, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (23, 4, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (32, 4, 5)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1032, 5, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1034, 5, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1035, 5, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1036, 5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1037, 5, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1040, 6, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1041, 5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1043, 4, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1044, 1, 5)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1045, 2, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1046, 4, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1047, 5, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1049, 5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1050, 5, 3)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1051, 5.5, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1054, 5.5, 4)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1057, 6, 2)
```

```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1060, 5.5, 4)
```



```
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1061, 4.7, 2)
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1062, 4.7, 2)
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1063, 5.5, 3)
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1064, 5.5, 3)
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1065, 5.5, 3)
INSERT INTO [dbo].[Ecra] ([idEcra], [tamanho], [idResoluçãoEcra]) VALUES (1066, 5.5, 2)
SET IDENTITY_INSERT [dbo].[Ecra] OFF
```

```
SET IDENTITY_INSERT [dbo].[Estado] ON
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (1, N'Péssimo')
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (2, N'Mau')
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (3, N'Intermédio')
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (4, N'Bom')
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (5, N'Muito Bom')
INSERT INTO [dbo].[Estado] ([idEstado], [designação]) VALUES (6, N'Como Novo')
SET IDENTITY_INSERT [dbo].[Estado] OFF
```

```
SET IDENTITY_INSERT [dbo].[marca] ON
INSERT INTO [dbo].[marca] ([idMarca], [nome]) VALUES (1, N'Apple')
INSERT INTO [dbo].[marca] ([idMarca], [nome]) VALUES (2, N'Samsung')
INSERT INTO [dbo].[marca] ([idMarca], [nome]) VALUES (3, N'LG')
INSERT INTO [dbo].[marca] ([idMarca], [nome]) VALUES (4, N'Outras')
SET IDENTITY_INSERT [dbo].[marca] OFF
```

```
SET IDENTITY_INSERT [dbo].[Processador] ON
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (15, 3, 8)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (16, 1.4, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (17, 2.7, 8)
```



```

INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (18, 2.5, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (19, 1, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (20, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (22, 1.3, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (23, 2, 2)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (32, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1032, 2, 8)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1034, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1035, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1036, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1037, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1040, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1041, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1043, 1, 1)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1044, 1, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1045, 2, 4)

```

```

INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1046, 1, 8)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1047, 1, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1049, 1, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1050, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1051, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1054, 2, 8)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1057, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1060, 2.4, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1061, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1062, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1063, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1064, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1065, 2, 4)
INSERT INTO [dbo].[Processador] ([idProcessador], [velocidadeProcessador],
[nucleosProcessador]) VALUES (1066, 1.4, 2)
SET IDENTITY_INSERT [dbo].[Processador] OFF

SET IDENTITY_INSERT [dbo].[ResoluçãoEcra] ON

```

```
INSERT INTO [dbo].[ResoluçãoEcra] ([idResoluçãoEcra], [designação]) VALUES (1, N'640x480 (VGA)')  
INSERT INTO [dbo].[ResoluçãoEcra] ([idResoluçãoEcra], [designação]) VALUES (2, N'1280x720 (HD)')  
INSERT INTO [dbo].[ResoluçãoEcra] ([idResoluçãoEcra], [designação]) VALUES (3, N'1920x1080 (FULL HD)')  
INSERT INTO [dbo].[ResoluçãoEcra] ([idResoluçãoEcra], [designação]) VALUES (4, N'2560x1440 (2K)')  
INSERT INTO [dbo].[ResoluçãoEcra] ([idResoluçãoEcra], [designação]) VALUES (5, N'3840x2160 (4K)')  
SET IDENTITY_INSERT [dbo].[ResoluçãoEcra] OFF
```